Programsko inženjerstvo

Ak. god. 2020./2021.

GeoFighter

Dokumentacija, Rev. 2.

Grupa: *Ferllowship*

Voditelj: *Matija Frandolić*

Datum predaje: *14. siječnja 2021.*

Nastavnik: *Hrvoje Nuić*

Sadržaj

1. Dnevnik promjena dokumentacije

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Napravljen predložak	Frandolić	07.10.2020.
0.2	Dodani uvod i razrada u opis projektnog zadatka	Kopić	17.10.2020.
0.3	Dodan ostatak opisa projektnoga zadatka (potencijalne koristi, korisnici i moguće nadogradnje)	Kopić	20.10.2020.
0.4	Dodani funkcionali zahtjevi	Bašić	20.10.2020.
0.5.1	Dodan prvi dio opisa obrazaca uporabe	Krivačić	20.10.2020.
0.5.2	Dodan drugi dio opisa obrazaca uporabe	Bašić Krivačić	21.10.2020.
0.6	Dodani dijagrami obrazaca uporabe	Krivačić	25.10.2020.
0.7	Dodani sekvencijski dijagrami	Bašić Krivačić	28.10.2020.
0.8	Dodana postojeća slična rješenja u Opis projektnoga zadatka	Kopić	29.10.2020.
0.9.1	Dodan opis i dijagrami baze podataka	Petrović Brečić	05.11.2020.
0.9.2	Izmjene u opisu baze podataka	Petrović Brečić	12.11.2020.
0.10.1	Dodani opis i prvi dijagrami razreda	Brečić Petrović	12.11.2020.
0.10.2	Izmjenjen opis arhitekture sustava i dovršeni dijagrami razreda	Brečić Krivačić Kopić	13.11.2020.
1.0	Pregledano prije predaje za 1. ciklus	Frandolić	13.11.2020.
1.1	Dodan predložak za drugu reviziju	Frandolić	30.12.2020.
1.2	Dodan dijagram stanja	Krivačić	11.1.2021.
1.3	Dodan dijagram razmještaja	Krivačić	12.1.2021.

Rev.	**Opis promjene/dodatka**	**Autori**	**Datum**
1.4	Dodano ispitivanje sustava	Krivačić	13.1.2021.
1.5	Dodan dijagram komponenti	Kopić	13.1.2021.
1.6	Ažurirani dijagrami razreda	Brečić	13.1.2021.
1.7	Dodan dijagram aktivnosti	Petrović	14.1.2021.
1.8	Napisano potpoglavlje 5.1 Korištene tehnologije i alati	Petrović	14.1.2021.
1.9	Dodano ispitivanje komponenti	Brečić	14.1.2021.
1.10	Dodano upute za puštanje u pogon	Frandolić	14.1.2021.
1.11	Dodano dijagrami pregleda promjena	Frandolić	14.1.2021.
2.0	Pregledano prije predaje za 2. ciklus	Frandolić	14.1.2021.

2. Opis projektnog zadatka

2.1 Uvod

Kroz ovaj je projekt potrebno razviti programsku potporu za igru *GeoFighter*. *GeoFighter* je web-aplikacija kojom se u obliku karata određenih jačina evidentiraju stvarne lokacije koje su korisnici (č. igrači) posjetili. Skupljenim se kartama tada geografski bliski igrači mogu međusobno boriti. Pobjednik borbe dobiva određeni broj bodova koji se oduzimaju gubitniku te se promjenom bodova mijenja i globalni rang igrača. Cilj igrača je posjetiti što veći broj mjesta, skupiti što više karata i postići što bolji rang.

2.2 Razrada

2.2.1 Registracija i prijava

Prije korištenja same aplikacije, ukoliko je korisnik novi igrač, dužan je najprije registrirati se. Za registraciju potrebni su:

- korisničko ime,
- fotografija,
- lozinka,
- e-mail adresa.

Validacijom podataka utvrđuje se ispravnost unosa podataka te se provjerava postoji li već korisnik s istim korisničkim imenom ili e-mail adresom. Kada registracija bude gotova, korisniku se na navedenu adresu šalje poveznica putem koje potvrđuje svoj račun. Tek nakon što je račun potvrđen, omogućuje se ulazak u sâm igrački sustav. Ako je igrač već prethodno registriran, on u sustav ulazi upisujući korisničko ime i lozinku.

2.2.2 Korištenje sustava

Početni zaslon i mogućnosti

Pri ulasku u sustav korisniku se na karti prikazuje njegova lokacija zajedno s obližnjim lokacijama. Pritiskom na određenu lokaciju otvara se prozor s atributima poput naziva lokacije, opisa, fotografije, kategorije i jačina karte te lokacije. Kategorije lokacija uključuju naseljena mjesta, vrhove planina i gora, umjetničke instalacije... Na zaslonu se nalazi simbol za padajući izbornik. Pritiskom na taj simbol, izbornik se otvara na vrhu te prikazuje simbole koji označavaju:

- **Početna stranica** - korisnik ima mogućnost da se vrati na početni zaslon te da sudjeluje u borbi ili se dopisuje s aktivnim igračima.

- **Profil korisnika** - otvara novi prozor u kojem se prikazuju podatci o korisniku(korisničko ime, fotografija, e-mail adresa). Za fotografiju, e-mail i lozinku sa strane nudi se mogućnost uređivanja. Osim toga, ovaj prozor nudi mogućnost da se korisnik prijavi za ulogu kartografa.

- **Kolekciju korisnikovih karata** - otvara novi prozor unutar kojega su prikazane karte koje je korisnik skupio. Karte je zatim moguće poredati prema kategorijama ili prema jačini.

- **Globalnu rang ljestvicu** - otvara novi prozor koji prikazuje redne brojeve igrača, njihova korisnička imena i brojeve bodova poredane silazno.

- **Pomoć korisniku** - otvara prozor unutar kojeg korisnik može postaviti pitanje administratoru.

Izbornik se zatvara ponovnim klikom na njegov simbol te se vraća početni zaslon. Igrač zatim svoje stvarno kretanje prati na karti te pokušava obići što više stvarnih lokacija. Kada igrač stigne na neku lokaciju koja je označena kao karta, pojavljuje se poruka koja javlja da korisnik ima novu lokaciju u kolekciji karata te prikazuje atribute te lokacije.

Borba

Ukoliko korisnik ugleda na karti u krugu od 50 km drugoga igrača s kojim se želi boriti, pritiskom na drugog igrača pojavljuje se prozor u kojemu onda potvrđuje slanje pozivnice za borbu ili odustaje od izazova. S druge strane, drugome se igraču prikazuje iskočni prozor s ponudom za borbu i podatcima o protivniku kao što su korisničko ime, broj bodova i trenutačni rang. Tada drugi igrač može prihvatiti ili odbiti borbu.

Ako je borba prihvaćena, svaki od igrača bira po tri karte s kojima će se boriti. Borba se odvija u tri runde. Borbu prvi započinje igrač s manjim globalnim rangom. On odabire jednu od svojih karata, koje je prethodno odabrao i baca ju. Drugi igrač zatim također odabire kartu i baca ju. Ukoliko su karte jednake jačine, poništava se runda. Jača karta pobjeđuje te sljedeću rundu otvara pobjednik prethodne runde. Igrač koji ima najviše bodova nakon odigranih rundi je pobjednik. Tada pobjednik dobiva broj bodova koji odgovara broju bodova najjače karte, dok se isti taj broj bodova oduzima gubitniku. Karte korištene u borbi zatim slabe za broj bodova koji imaju podijeljen sa 100. Prozor s borbom se zatvara kod oba igrača te im se prikazuje ažuriranje njihovih osobnih bodova i bodova karata iz kojega se zatim vraća na početni zaslon.

Bodovanje karata

Svaka karta ima broj bodova koji mu daje kategorija kojoj pripada. Što je kategorija rjeđa i teža za posjetiti (npr. vrh planine naspram nekoga grada) to donosi više bodova. Svaka karta osim toga ima i početnih 100 bodova koji se onda smanjuju za 0.25 boda za svaku posjetu toj lokaciji. Minimalni broj tih bodova 0. Ukupan broj bodova neke karte je zbroj tih dvaju vrijednosti.

Naziv kategorije	**Broj bodova**
Grad	25
Naselje	30
Umjetnička instalacija	35
Vrh planine	40

2.2.3 Održavanje sustava

Administratori

Administratori imaju sve ovlasti kao i ostali registrirani korisnici uz još neke dodatne. Oni imaju pristup bazi podataka i mogućnost uređivanja profila korisnika. Administratori također pregledavaju prijave za kartografe i odobravaju ih ili odbijaju. Osim toga, ukoliko korisnici pošalju pitanja unutar prozora za pomoć, oni na njih odgovaraju.

Kartografi

Ako korisniku administrator odobri status kartografa, prije nego što počne koristiti svoje kartografske ovlasti, on mora u svoje korisničke podatke dodati IBAN računa za uplatu i fotografiju osobne iskaznice. Kartografi su osobe zadužene za nadopunjavanje baze podataka s lokacijama koje su igrači prijavili za moguće proširenje. Njima se u padajućem izborniku pojavljuje posebni gumb pomoću kojega se otvara prozor s prijedlozima za nove lokacije. Osim toga, prijedlozi im se prikazuju na karti, a oni ih mogu odbiti, potvrditi, urediti ili označiti da je potrebna potvrda s terena. Kartograf za one prijave za koje je potrebno potvrđivanje s terena može označiti da će ih osobno doći provjeriti, a sustav mu preko vanjskog servisa OSRM dohvaća najbliži put do lokacije.

2.3 Potencijalne koristi

Geofighter potiče igrače na istraživanje onih manje poznatih ili čak neotkrivenih lokacija oko njih. Osim toga, korisnika se obogaćuje novim znanjima pomoću informacija o lokacijama na kartama i motivira na otkrivanje novih kultura što dovodi do razvijanja međuljudskih odnosa.

2.4 Korisničke skupine

Ova je aplikacija prije svega namijenjena ljudima natjecateljskoga duha koji uz to vole i putovati ili jednostavno onima koji žele naučiti nove informacije o već poznatim lokacijama kroz igru. Igrači bi trebali biti boljeg fizičkog stanja kako bi došli do teže dostupnih lokacija te biti punoljetne kako bi se mogle kretati bez većih zakonskih ograničenja. Skupine igrača koje najviše odgovaraju opisu su osobe od oko

20 do oko 50 godina, neovisno kojega spola.

2.5 Moguće nadogradnje

Kao moguće nadogradnje naveli bismo:

- **Prozor za razgovor između igrača** - igrači unutar 50 km imali bi mogućnosti jedni drugima poslati poruke tijekom istraživanja karte i tijekom borbe.

- **Organizacija turnira** - unutar igre bi se organizirali turniri nakon kojih bi pobjednici dobivali određeni broj bodova ili, u daljoj budućnosti i u potencijalnoj suradnji sa zainteresiranim turističkim agencijama koje bi zauzvrat dobivale reklame, putovanja na neke od lokacija na kartama.

- **Proširenje udaljenosti za borbu** - udaljenost od 50 km unutar koje se igrači mogu boriti bi se potencijalno mogla povećati.

2.6 Postojeća slična rješenja

GeoFighter bi se mogao definirati kao hibrid između igara proširene stvarnosti u kojima korisnici putuju stvarnim lokacijama kako bi pristupili lokacijama, artifaktima i sl. unutar igre poput: Harry Potter: Wizards Unite, Pokemon GO, Ingress, The Walking Dead: Our World... i igara u kojima igrači skupljaju karte kao što su: Faeria, Hearthstone, Magic: The Gathering Arena... Kao sustave s najsličnijim specifikacijama treba izdvojiti Pokemon GO - igru u kojoj igrači na stvarnim lokacijama skupljaju Pokemone raznih rijetkosti, posjećuju tzv. Pokemon dvorane u kojima se mogu boriti s drugim stvarnim igračima te skupljati bodove i tako dalje napredovati. Pokemon GO kao framework aplikacije koristi Libgdx i programske jezike Javu, C++ i C#. Osim Pokemon GO-a trebalo bi izdvojiti i Hearthstone - igru u kojoj igrači skupljaju karte iz različitih kategorija u dekove kako bi se zatim njima borili s drugim igračima. Hearthstone je rađen pomoću Unity-ja, pokretača igara za više platformi, u C#.

3. Specifikacija programske potpore

3.1 Funkcionalni zahtjevi

Dionici:

1. Igrači
2. Kartografi
3. Administrator
4. Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

1. Neregistrirani/neprijavljeni korisnik (inicijator) može:
   (a) obaviti registraciju:
      i. kao igrač unosom korisničkog imena, fotografije, e-mail adrese i lozinke
      ii. kao kartograf unosom korisničkog imena, fotografije, e-mail adrese, lozinke, IBAN-a te fotografije osobne iskaznice
   (b) ukoliko je korisnik već registriran u sustavu, mora se prijaviti koristeći korisničko ime ili e-mail adresu i lozinku
2. Igrač (inicijator) može:
   (a) pregledavati i mijenjati svoje korisničke podatke
   (b) izbrisati svoj korisnički račun
   (c) vidjeti kartu na kojoj su označene lokacije na kojima se mogu skupiti karte te njegova lokacija
   (d) vidjeti informacije o lokacijama (naziv, opis, fotografiju, kategoriju i jačinu karte te lokacije)
   (e) vidjeti kolekciju svojih karata
   (f) vidjeti popis ostalih aktivnih igrača koji se nalaze u krugu od 50km i s njima ući u borbu
   (g) prijaviti novu lokaciju

(h) vidjeti globalnu statistiku odigranih borbi i sakupljenih lokacija
(i) vidjeti poredak svih igrača
(j) vidjeti profil drugog igrača (njegove karte, rang na globalnoj ljestvici i statistike vezane uz zadnjih 10 borbi s drugim igračima)

3. Kartograf (inicijator) može:

   (a) vidjeti kartu sa:
      i. prijavljenim lokacijama za unos u bazu podataka
      ii. prikazanim najkraćim putem do lokacije koju treba provjeriti na terenu
      iii. već unesenim lokacijama u bazi podataka
   (b) odbiti, potvrditi, urediti ili označiti da je potrebna potvrda s terena za prijavljenu lokaciju
   (c) dodavati lokacije u bazu podataka
   (d) vidjeti i izmjeniti svoje osobne podatke

4. Administrator (inicijator) može:

   (a) vidjeti i uređivati popis svih korisnika i njihovih osobnih podataka
   (b) dodjeliti igračima privremeno isključenje iz igre
   (c) vidjeti i uređivati postojeće lokacije
   (d) vidjeti globalnu statistiku odigranih borbi i sakupljenih lokacija
   (e) vidjeti poredak svih igrača

5. Baza podataka (sudionik):

   (a) pohranjuje sve podatke o korisnicima i njihovim ovlastima
   (b) pohranjuje sve podatke o kartama i lokacijama
   (c) pohranjuje sve podatke o borbama i rang listama igrača

6. OSRM (sudionik):

   (a) prikazuje kartografu na karti najkraći put (problem trgovačkog putnika) do lokacije koju treba provjeriti

3.1.1 Obrasci uporabe

Opis obrazaca uporabe

UC1 - Početni zaslon za neprijavljenog korisnika

- **Glavni sudionik:** Korisnik
- **Cilj:** Prikaz početnog zaslona za neprijavljenog korisnika
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
  1. Korisnik pristupa aplikaciji
  2. Otvara se početni zaslon za neprijavljenog korisnika

UC2 -Registracija igrača

- **Glavni sudionik:** Korisnik
- **Cilj:** Stvoriti korisnički račun za pristup sustavu
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za registraciju igrača
  2. Korisnik unosi potrebne korisničke podatke(korisničko ime, e-mail, lozinka, fotografija)
  3. Korisnik prima obavijest o uspješnoj registraciji
  4. Nakon registracije korisnik automatski prijavljen u sustav
- **Opis mogućih odstupanja:**
  - Odabir već zauzetog korisničkog imena i/ili e-maila ili pružanje neispravnog e-maila
    1. Sustav obavještava korisnika o neuspjelom upisu i vraća ga na stranicu za registraciju
    2. Korisnik mijenja potrebne podatke te završava unos ili odustaje od registracije
  - Odabir ”slabe” lozinke(za ”jaku” lozinku obavezno barem 8 znakova)
    1. Sustav obavještava korisnika da je lozinka ”slaba” i vraća ga na stranicu za registraciju
    2. Korisnik odabire ”jaču” lozinku te završava unos ili odustaje od registracije

UC3 -Prijava u sustav

- **Glavni sudionik:** Igrač
- **Cilj:** Dobiti pristup korisničkom sučelju
- **Sudionici:** Baza podataka
- **Preduvjet:** Registracija
- **Opis osnovnog tijeka:**
    1. Unos korisničkog imena i lozinke
    2. Potvrda o ispravnosti unesenih podataka
    3. Pristup korisničkim funkcijama
- **Opis mogućih odstupanja:**
    - Unos neispravnog korisničkog imena ili lozinke
        1. Sustav obavještava korisnika o neispravnom upisu i vraća ga na stranicu za prijavu

UC4 - Prijava za kartografa

- **Glavni sudionik:** Igrač
- **Cilj:** Proširenje ovlasti igrača
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
    1. Korisnik odabire opciju prijava za kartografa
    2. Korisnik unosi potrebne korisničke podatke(IBAN računa za uplatu i fotografiju osobne iskaznice)
    3. Korisnik prima obavijest o uspješnoj prijavi za kartografa te čeka odobrenje administratora
- **Opis mogućih odstupanja:**
    - Neispravan unos IBAN-a
        1. Sustav obavještava korisnika o pogrešnom unosu i vraća ga na stranicu za prijavu
        2. Korisnik mijenja potrebne podatke te završava prijavu ili odustaje od prijave

UC5 - Skupljanje karata

- **Glavni sudionik:** Igrač
- **Cilj:** Igrač skupi karte
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju pregled lokacija na karti
  2. Sustav otvara kartu s lokacijama
  3. Korisnik odabire lokaciju
  4. Ako je korisnik skupio kartu prikažu se podaci o karti, inače prikazuje se opcija da skupi karticu te se dodaje u kolekciju

UC6 -Pregled informacija o lokaciji

- **Glavni sudionik:** Igrač
- **Cilj:** Pregled informacija o lokaciji
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju ”Informacija o lokaciji”
  2. Aplikacija prikazuje informacije o lokaciji

UC7 -Pregled osobnih podataka igrača

- **Glavni sudionik:** Igrač
- **Cilj:** Pregledati osobne podatke
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju ”Osobni podatci”
  2. Aplikacija prikazuje osobne podatke korisnika

UC8 -Promjena osobnih podataka igrača

- **Glavni sudionik:** Igrač
- **Cilj:** Promjeniti osobne podatke
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Igrač je prijavljen

- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za promjenu podataka
  2. Korisnik mijenja svoje osobne podatke
  3. Korisnik sprema promjene
  4. Baza podataka se ažurira
- **Opis mogućih odstupanja:**
  - Odabir već zauzetog korisničkog imena i/ili e-maila
    1. Sustav obavještava korisnika da je korisničko ime i/ili e-mail već zauzeto i traži ponovni unos
    2. Korisnik unosi novo korisničko ime i/ili e-maila ili odustaje je promjene osobnih podataka
  - Korisnik odabire ”slabu” zamjensku lozinku
    1. Sustav obavještava korisnika da je odabrao ”slabu” lozinku i traži ponovni unos
    2. Korisnik unosi novu lozinku koja je dovoljno ”jaka” ili odustaje od promjene lozinke
  - Korisnik promijeni svoje osobne podatke, ali ne odabere opciju ”Spremi promjenu”
    1. Sustav obavještava korisnika da nije spremio podatke prije izlaska iz prozora

UC9 - Pregled osobnih podataka kartografa

- **Glavni sudionik:** Kartograf
- **Cilj:** Pregledati osobne podatke
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Kartograf je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju ”Osobni podatci”
  2. Aplikacija prikazuje osobne podatke korisnika

UC10 -Promjena osobnih podataka kartografa

- **Glavni sudionik:** Kartograf
- **Cilj:** Promjeniti osobne podatke
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Kartograf je prijavljen
- **Opis osnovnog tijeka:**

1. Korisnik odabire opciju za promjenu podataka
2. Korisnik mijenja svoje osobne podatke
3. Korisnik sprema promjene
4. Baza podataka se ažurira

- **Opis mogućih odstupanja:**
  - Odabir već zauzetog korisničkog imena i/ili e-maila
    1. Sustav obavještava korisnika da je korisničko ime i/ili e-mail već zauzeto i traži ponovni unos
    2. Korisnik unosi novo korisničko ime i/ili e-maila ili odustaje je promjene osobnih podataka
  - Korisnik odabire ”slabu” zamjensku lozinku
    1. Sustav obavještava korisnika da je odabrao ”slabu” lozinku i traži ponovni unos
    2. Korisnik unosi novu lozinku koja je dovoljno ”jaka” ili odustaje od promjene lozinke
  - Korisnik promijeni svoje osobne podatke, ali ne odabere opciju ”Spremi promjenu”
    1. Sustav obavještava korisnika da nije spremio podatke prije izlaska iz prozora

UC11 -Brisanje korisničkog računa

- **Glavni sudionik:** Igrač, kartograf
- **Cilj:** Izbrisati svoj korisnički račun
- **Sudionici:** Baza podataka
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik otvara stranicu s osobnim podacima
  2. Korisnik briše račun
  3. Sustav traži potvrdu brisanja korisničkog računa
  4. Korisnik odobrava brisanje
  5. Korisnički račun se izbriše iz baze podataka
  6. Otvara se stranica za registraciju

UC12 -Pregled (vlastith) sakupljenih karata

- **Glavni sudionik:** Igrač, administrator
- **Cilj:** Pregled skupljenih karata igrača
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za pregled skupljenih karata
  2. Otvara se stranica sa skupljenim kartama korisnika
- **Opis mogućih odstupanja:**
  - Korisnik nije skupio niti jednu kartu
    1. Sustav obavještava korisnika da mora skupiti barem jednu kartu da može pristupiti pregledu i vraća ga na kartu

UC13 -Pregled vlastite statistike igrača

- **Glavni sudionik:** Igrač
- **Cilj:** Pregledati vlastitu statistiku
- **Sudionici:** Baza podataka
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju ”Moja statistika”
  2. Aplikacija prikazuje vlastitu statistiku igrača

UC14 - Pregled (vlastith) odigranih borbi

- **Glavni sudionik:** Igrač
- **Cilj:** Pregled borbi u kojima je igrač sudjelovao
- **Sudionici:** Baza podataka
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za pregled odigranih borbi
  2. Otvara se stranica s odigranim borbama
- **Opis mogućih odstupanja:**
  - Korisnik nije sudjelovao u niti jednoj borbi
    1. Sustav obavještava korisnika da nije sudjelovao u borbi

UC15 -Pregled vlastite statistike kartografa

- **Glavni sudionik:** Kartograf
- **Cilj:** Pregledati vlastitu statistiku
- **Sudionici:** Baza podataka
- **Preduvjet:** Kartograf je prijavljen
- **Opis osnovnog tijeka:**
    1. Korisnik odabire opciju ”Moja statistika”
    2. Aplikacija prikazuje vlastitu statistiku kartografa

UC16 - Pregled profila ostalih igrača

- **Glavni sudionik:** Igrač
- **Cilj:** Pregled profila ostalih igrača
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
    1. Korisnik odabire profil drugog igrača
    2. Otvara se profil drugog igrača

UC17 -Pregled statistike ostalih igrača

- **Glavni sudionik:** Igrač
- **Cilj:** Pregledati statistiku ostalih igrača
- **Sudionici:** Baza podataka
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
    1. Korisnik odabire opciju ”Statistika”
    2. Aplikacija prikazuje statistiku igrača

UC18 -Pregled sakupljenih karata ostalih igrača

- **Glavni sudionik:** Igrač
- **Cilj:** Pregled skupljenih karata igrača
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
    1. Korisnik odabire opciju za pregled skupljenih karata
    2. Otvara se stranica sa skupljenim kartama korisnika

- **Opis mogućih odstupanja:**
  - Korisnik nije skupio niti jednu kartu
    1. Sustav obavještava korisnika da igrač nije sakupio niti jednu kartu

UC19 - Izazivanje na borbu igrača u blizini

- **Glavni sudionik:** Igrač
- **Cilj:** Izazvati ostale igrače na borbu
- **Sudionici:** Baza podataka
- **Preduvjet:** Igrač je prijavljen i sakupio barem tri karte
- **Opis osnovnog tijeka:**
  1. Korisnik odabire igrača s kojim se želi boriti
  2. Korisnik odabire opciju ”Izazovi”
  3. Korisnik čeka odgovor izazvanog igrača
- **Opis mogućih odstupanja:**
  - Izazvani igrač nije skupio barem tri karte
    1. Sustav obavještava korisnika da željeni igrač nije skupio barem tri karte te ne može sudjelovati u borbi

UC20 - Odluka o prihvaćanju izazova

- **Glavni sudionik:** Igrač
- **Cilj:** Igrač može prihvatiti ili odbiti borbu
- **Sudionici:** Baza podataka
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za prihvaćanje ili odbijanje izazova
  2. Ako je prihvatio izazov započinje borba te se otvara stranica za borbu igrača

UC21 - Borba igrača

- **Glavni sudionik:** Igrač
- **Cilj:** Borba igrača za najbolji globalni rang
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Igrač odabire 3 karte iz svoje kolekcije s kojima želi ući u borbu te čeka da suparnik učini isto

2. Igrač koji ima manji globalni rang igra prvi
3. Igrač izvlači jednu od tri izabrane karte, na što suparnik odgovara izvlačenjem jedne od svojih karata
4. Sustav uspoređuje dvije karte te igraču s jačom kartom upisuje bod
5. Igrač koji pobijedi u rundi prvi baca kartu u idućoj
6. Borba traje dok igrači ne iskoriste sve izabrane karte
7. Igrač s više bodova pobjeđuje u borbi

- **Opis mogućih odstupanja:**
  - Karte koje se uspoređuju su jednake jačine
    1. Sustav obavještava korisnike da je runda poništena
    2. Borba se nastavlja
  - Korisnici imaju jednak broj bodova
    1. Sustav obavještava da je borba izjednačena

UC22 - Pregled globalnog ranga igrača

- **Glavni sudionik:** Igrač
- **Cilj:** Pregled globalnog ranga igrača
- **Sudionici:** Baza podataka
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za pregled globalnog ranga igrača
  2. Otvara se stranica s globalnim rangom igrača

UC23 - Globalna statistika odigranih borbi

- **Glavni sudionik:** Igrač
- **Cilj:** Globalna statistika odigranih borbi
- **Sudionici:** Baza podataka
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za pregled globalne statistike odigranih borbi
  2. Otvara se stranica s globalnom statistikom odigranih borbi

UC24 - Globalna statistika sakupljenih lokacija

- **Glavni sudionik:** Igrač
- **Cilj:** Globalna statistika sakupljenih lokacija
- **Sudionici:** Baza podataka

- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za pregled globalne statistike sakupljenih lokacija
  2. Otvara se stranica s globalnom statistikom sakupljenih lokacija

UC25 - Prijava nove lokacije

- **Glavni sudionik:** Igrač
- **Cilj:** Dodati novu lokaciju
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Igrač je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire lokaciju koju želi dodati
  2. Korisnik dodaje karakteristike lokacije

UC26 - Verificiranje prijavljene lokacije

- **Glavni sudionik:** Kartograf
- **Cilj:** Verificirati prijavljenu lokaciju
- **Sudionici:** Baza podataka
- **Preduvjet:** Kartograf je prijavljen
- **Opis osnovnog tijeka:**
  1. Kartograf potvrđuje ispravnost lokacije
  2. Baza podataka se ažurira

UC27 - Brisanje lokacije

- **Glavni sudionik:** Kartograf, administrator
- **Cilj:** Obrisati nepostojeću/neispravnu lokaciju
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik prijavljen i dodijeljena su mu prava administratora ili kartografa
- **Opis osnovnog tijeka:**
  1. Korisnik odabire lokaciju koju želi obrisati
  2. Baza podataka se ažurira

UC28 - Pregled svih prijavljenih(neobrađenih) lokacija

- **Glavni sudionik:** Kartograf
- **Cilj:** Pregled prijavljenih lokacija koje nisu obrađene
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Kartograf je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju pregled svih prijavljenih lokacija
  2. Otvara se popis svih prijavljenih lokacija koje nisu obrađene

UC29 - Karta s prikazanim najkraćim putem do lokacije za provjeru

- **Glavni sudionik:** Kartograf
- **Cilj:** Prikaz karte s prikazanim najkraćim putem do lokacije za provjeru
- **Sudionici:** Baza podataka, OSRM
- **Preduvjet:** Kartograf je prijavljen
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju karta s prikazanim najkraćim putem do lokacije za provjeru
  2. Otvara se karta

UC30 - Uređivanje postojeće lokacije

- **Glavni sudionik:** Kartograf, administrator
- **Cilj:** Promjeniti podatke o lokaciji
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Korisnik je prijavljen i dodijeljena su mu prava administratora ili kartografa
- **Opis osnovnog tijeka:**
  1. Korisnik odabire lokaciju za koju želi promijeniti podatke
  2. Korisnik mijenja podatke
  3. Korisnik sprema promjene
  4. Baza podataka se ažurira
- **Opis mogućih odstupanja:**
  - Kartograf promijeni podatke lokacije, ali ne odabere opciju ”Spremi promjenu”
    1. Sustav obavještava kartografa da nije spremio podatke prije izlaska iz prozora

UC31 - Pregled svih odigranih borbi

- **Glavni sudionik:** Administrator
- **Cilj:** Pregled svih odigranih borbi
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
  1. Administratora odabire opciju za pregled svih odigranih borbi
  2. Otvara se stranica s prikazom svih odigranih borbi

UC32 - Pregled korisnika

- **Glavni sudionik:** Administrator
- **Cilj:** Pregledati registrirane korisnike
- **Sudionici:** Baza podataka, Cloudinary
- **Preduvjet:** Korisnik je prijavljen i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
  1. Administrator odabire opciju pregledavanja korisnika
  2. Prikaže se lista svih ispravno registriranih korisnika s osobnim podacima

UC33 - Brisanje korisnika

- **Glavni sudionik:** Administrator
- **Cilj:** Obrisati korisnika
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
  1. Administrator odabire opciju uklanjanja korisnika
  2. Administrator pronalazi željenog korisnika
  3. Administrator uklanja željenog korisnika i njegove podatke iz baze podataka
  4. Baza podataka se ažurira

UC34 - Promjena prava pristupa

- **Glavni sudionik:** Administrator
- **Cilj:** Promijeniti razinu pristupa korisnika (igrač, kartograf, administrator)
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
    1. Administrator odabire opciju promjene prava pristupa
    2. Administrator pronalazi željenog korisnika
    3. Administrator mijenja razinu pristupa željenom korisniku

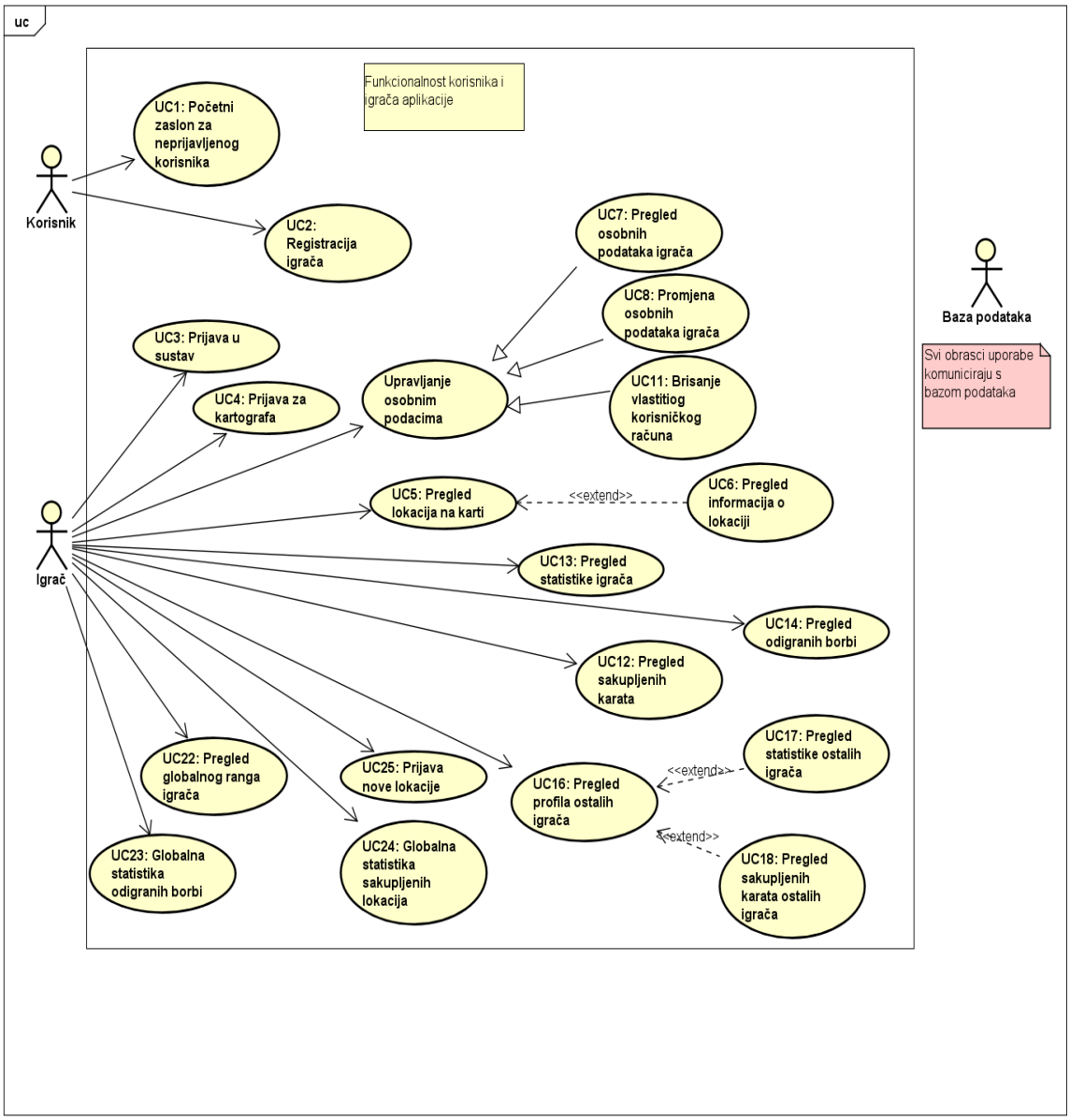
UC35 - Privremeno blokiranje korisnika

- **Glavni sudionik:** Administrator
- **Cilj:** Privremeno blokirati korisnika
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
    1. Administrator odabire opciju privremenog blokiranja korisnika
    2. Administrator pronalazi željenog korisnika
    3. Administrator privremeno blokira željenog korisnika

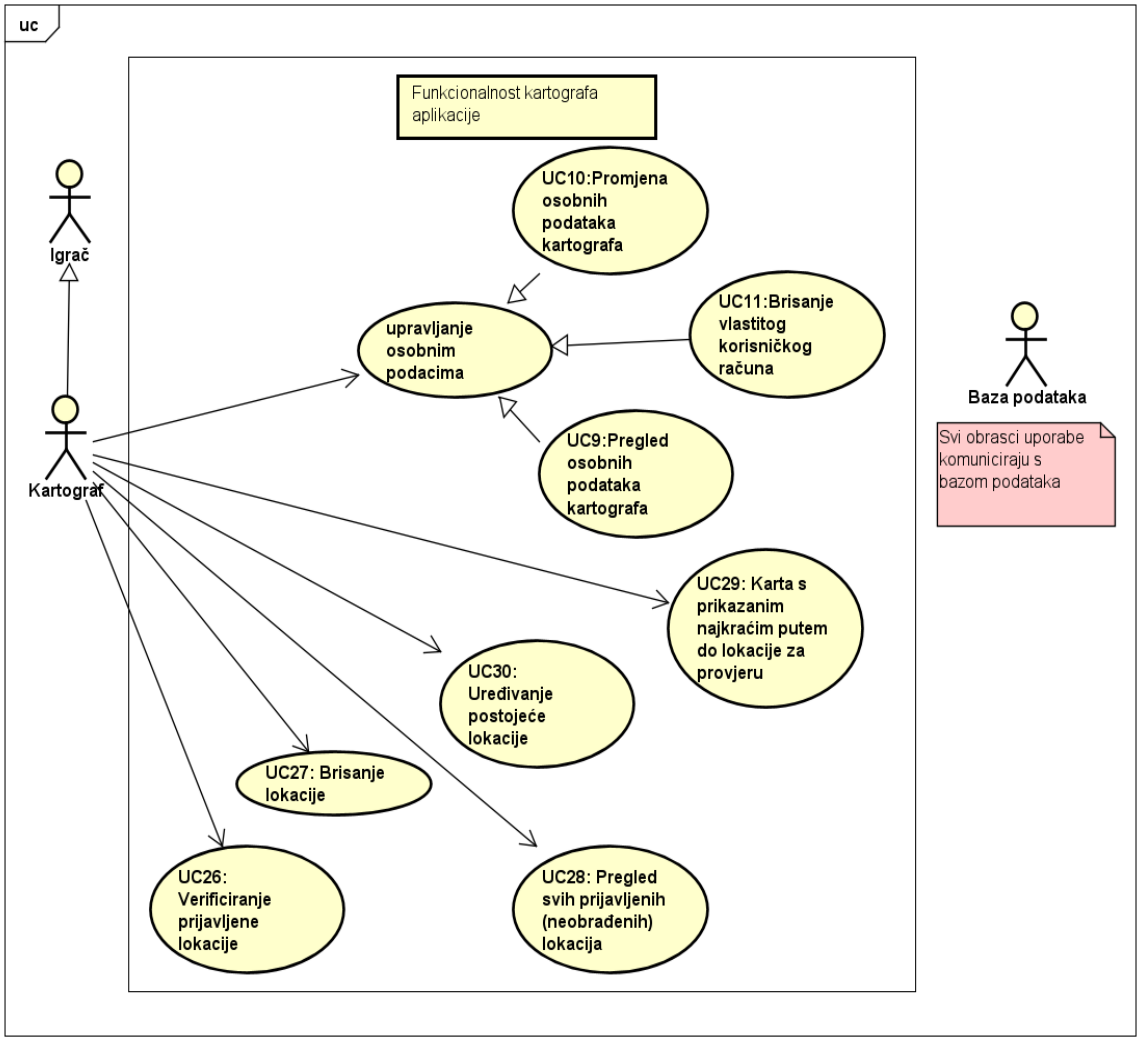
UC36 - Potvrda zahtjeva za kartografa

- **Glavni sudionik:** Administrator
- **Cilj:** Odobriti ili odbiti zahtjev igrača da postane kartograf
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
    1. Administrator odobri ili odbije zahtjev igrača
    2. Baza podataka se ažurira

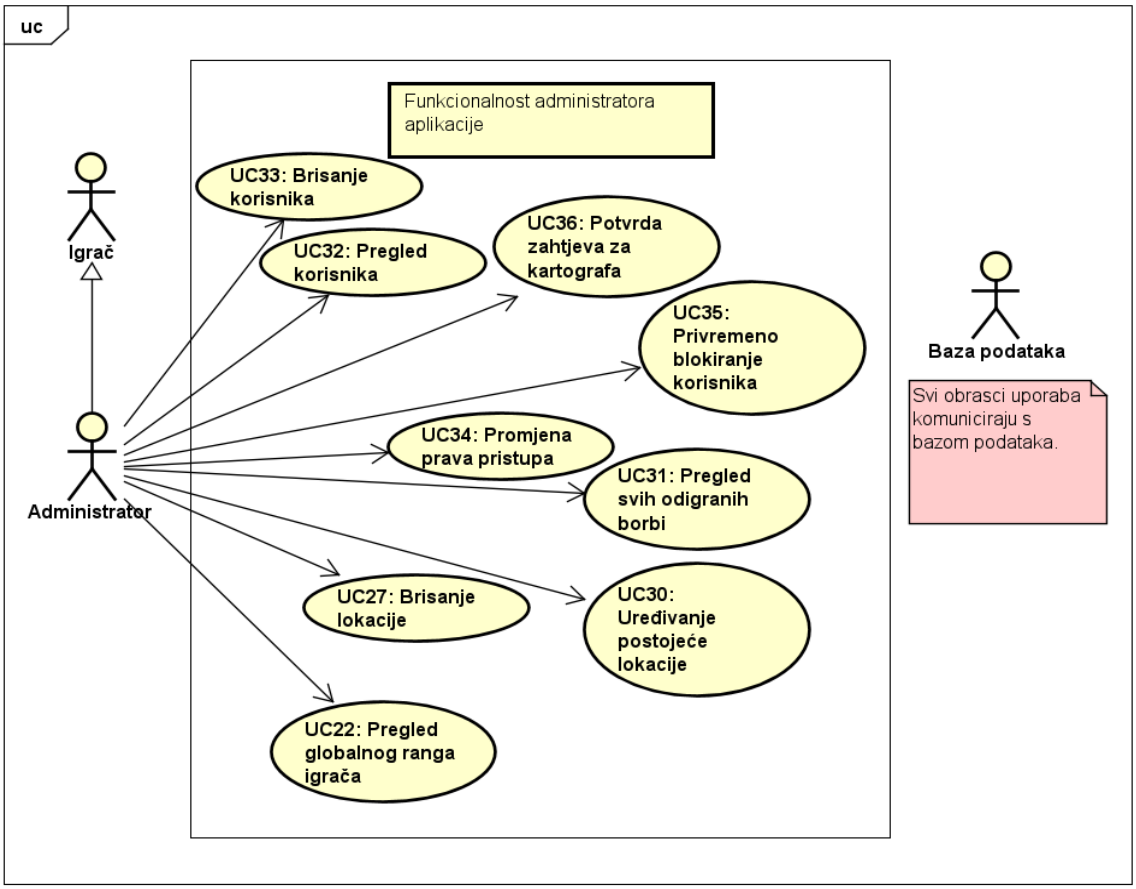
Dijagrami obrazaca uporabe



Slika 3.1: Dijagram obrasca uporabe, funkcionalnost korisnika i igrača



Slika 3.2: Dijagram obrasca uporabe, funkcionalnost kartografa

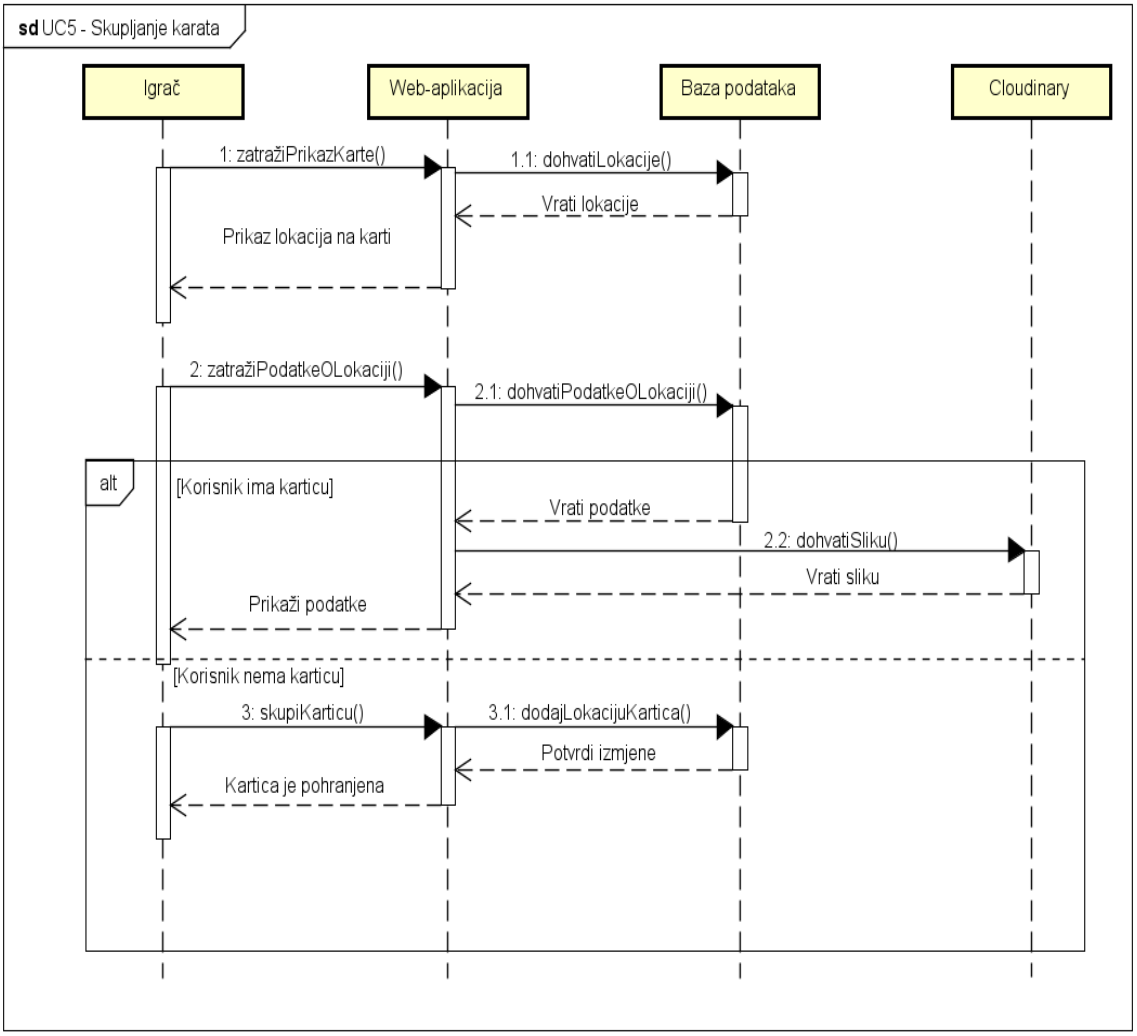


Slika 3.3: Dijagram obrasca uporabe, funkcionalnost administratora

3.1.2 Sekvencijski dijagrami

Obrazac uporabe UC5 - Skupljanje karata

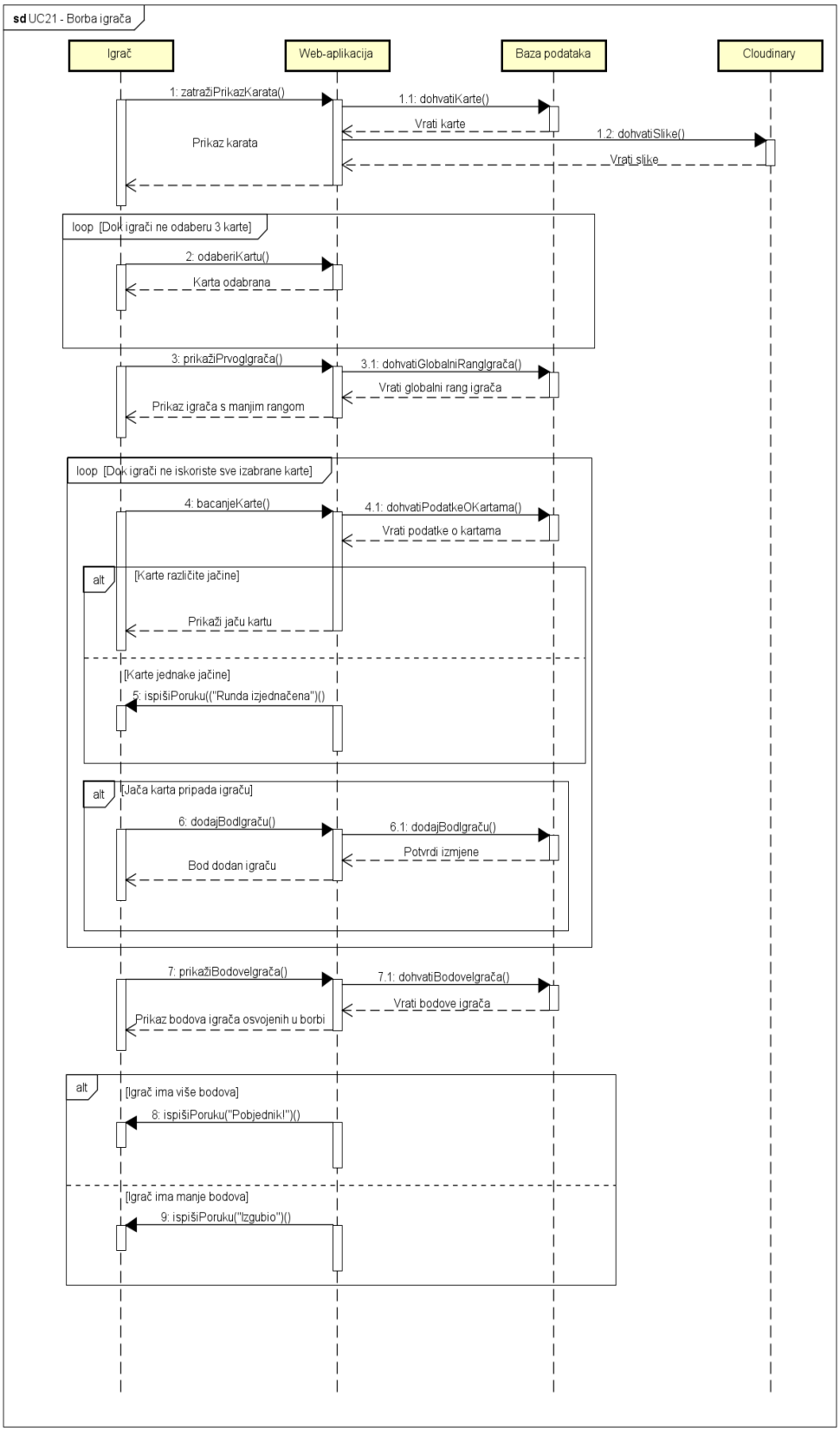
Igrač šalje zahtjev za prikaz karte s lokacijama kako bi mogao odabrati lokaciju koju želi skupiti. Poslužitelj dohvaća lokacije i prikazuje ih. Odabirom lokacije, poslužitelj iz baze podataka dohvaća osnovne podatke o lokaciji i prikazuje ih klijentu. Ako igrač nije skupio karticu, prikazuje mu se opcija da je sakupi. Poslužitelj tu informaciju prosljeđuje bazi koja spremna promjenu.



Slika 3.4: Sekvencijski dijagram UC5

Obrazac uprabe UC21 - Borba igrača

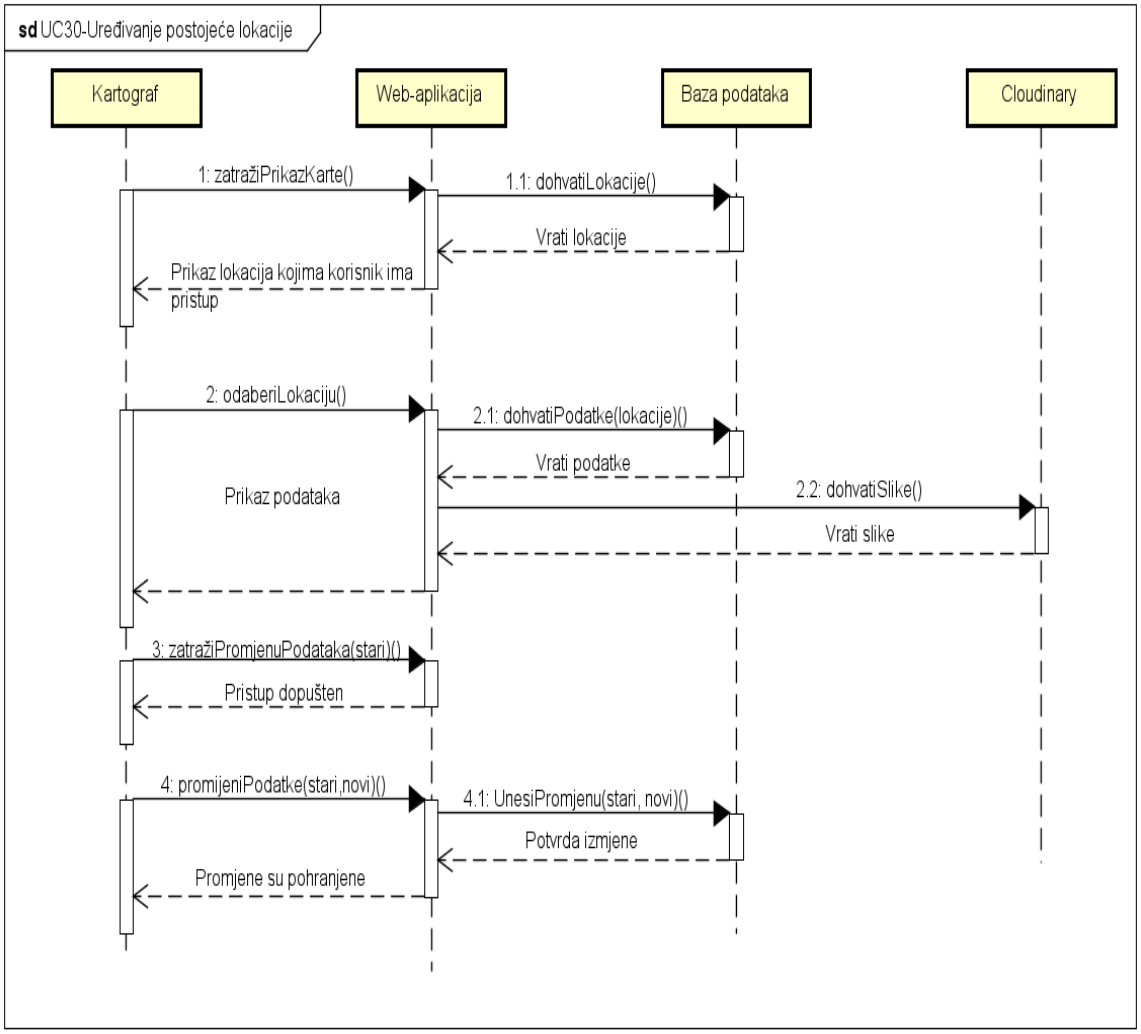
Igrač odabire opciju ”Prikaz karata”. Poslužitelj dohvaća karte iz baze podataka te ih prikazuje igraču. Igrači biraju 3 karte s kojima će se boriti. Na početku borbe prikazuje se koji igrač igra prvi. Poslužitelj dohvaća globalni rang igrača iz baze podataka te prikazuje igrača koji ima manji globalni rang. Dok ne iskoriste sve 3 izabrane karte, igrači bacaju naizmjenično kartu. Poslužitelj uspoređuje karte te vraća jaču kartu i dodaje bod igraču čija je to karta. Ukoliko su jačine karata jednake, sustav ispiše poruku da je runda izjednačena i nastavlja se borba. Na kraju borbe prikazuje se konačan broj bodova igrača. Igrač koji ima više bodova je pobjednik.



Slika 3.5: Sekvencijski dijagram UC21

Obrazac uporabe UC30 - Uređivanje postojeće lokacije

Kartograf odabire opciju ”Prikaz karte”. Poslužitelj dohvaća popis lokacija iz baze podataka te ih prikazuje kartografu na karti. Kartograf odabire lokaciju. Poslužitelj dohvaća podatke o lokaciji iz baze podataka te ih prikazuje kartografu.Kartograf šalje zahtjev za promjenu podataka o lokaciji. Poslužitelj mu daje pristup i kartograf unosi nove podatke o željenoj lokaciji. Poslužitelj izmijeni podatke u bazi podataka koja vraća potvrdu.



Slika 3.6: Sekvencijski dijagram UC30

3.2 Ostali zahtjevi

- Sustav treba omogućiti rad više korisnika u stvarnom vremenu
- Korisničko sučelje i sustav moraju podržavati hrvatsku abecedu (dijakritičke znakove) pri unosu tekstualnog sadržaja
- Izvršavanje dijela programa u kojem se pristupa bazi podataka ne smije trajati duže od nekoliko sekundi
- Sustav treba biti implementiran kao web aplikacija koristeći objektno-orijentirane jezike
- Neispravno korištenje korisničkog sučelja ne smije narušiti funkcionalnost i rad sustava
- Sustav treba biti jednostavan za korištenje, korisnici se moraju znati koristiti sučeljem bez opširnih uputa
- Nadogradnja sustava ne smije narušavati postojeće funkcionalnosti sustava
- Veza s bazom podataka mora biti kvalitetno zaštićena, brza i otporna na vanjske greške
- Pristup sustavu mora biti omogućen iz javne mreže pomoću HTTPS.

4. Arhitektura i dizajn sustava

Prilikom projektiranja samog sustava jedna od važnijih odluka bila je odabir programskih jezika i razvojnih okruženja u kojima ćemo razviti našu aplikaciju. Programski jezici koje smo odabrali su Java sa Spring Boot razvojnim okvirom za *backend* te React Javascript za *frontend*. Odabrana razvojna okruženja su Eclipse za *backend* te Visual Studio Code za *frontend*.

Također izbor odgovarajuće arhitekture programske potpore jedan je od najbitnijih koraka u oblikovanju sustava jer ona predstavlja poveznicu između zahtjeva u sustav i same implementacije sustava. Dobra arhitektura znači dobru fleksibilnost sustava, jednostavnu mogućnost nadogradnje i jeftino održavanje.

Sama arhitektura sustava je vrlo jednostavna, a sastoji se od dvije manje aplikacije: klijenta i poslužitelja. Osnovna prednost modela klijent-poslužitelj je u tome što nije potreban sustav za upravljanje bazom podataka na svakom klijentskom računalu, već se klijent s bazom podataka povezuje preko aplikacije. Time se postiže veća sigurnost i zaštita podataka. Poslužiteljska aplikacija ima pristup bazi podataka u kojoj će se pohranjivati podaci o lokacijama, kartama, borbama, igračima i kartografima. Poslužiteljska aplikacija neće dohvaćati slike i pohranjivati ih u bazu podataka već na servis Cloudinary. Poslužiteljska aplikacija temeljena je na REST principima za izradu web aplikacija te u skladu s time klijentska aplikacija dohvaća podatke iz poslužiteljske i prezentira ih na korisniku razumljiv način.

Za izradu aplikacije odabrali smo MVC arhitekturu jer omogućava dodatno strukturiranje aplikacije s obzirom na objektno orijentiranu paradigmu što znatno olakšava nezavisni razvoj, ispitivanje i održavanje aplikacije.

4.1 Baza podataka

Podatci potrebni za funkcioniranje naše aplikacije pohranjuju se u relacijsku bazu podataka. Osnovni objekt relacijske baze podataka je relacija - imenovana dvodimenzionalna tablica čiji stupci predstavljaju atribute, a retci opisuju entitete baze podataka (retci u relaciji zovu se n-torke). Sljedeći entiteti sačinjavaju bazu podataka naše aplikacije:

- Igrač (*player*)
- Zabrana pristupa (*ban*)
- Potvrda registracije (*confirmation*)
- Kartograf (*mapper*)
- Administrator (*admin*)
- Borba (*fight*)
- Lokacija (*location*)
- Najkraći put (*path*)
- Kategorija lokacije (*category*)
- Karta (*card*)

4.1.1 Opis tablica

player (Igrač) - Ovaj entitet sadržava podatke o korisniku. Svaki korisnik je ujedno i igrač. Sadrži atribute: ID korisnika, korisničko ime, hash lozinke, e-mail adresu, fotografiju, bodove igrača, status bana, aktivnost igrača, osposobljenost računa i "experience points". Ovaj entitet je u *One-to-Many* vezi s entitetom card preko atributa player_id. Ima dvije *One-to-Many* veze s entitetom fight, koje se odnose na borbe u kojima je igrač pobijedio i borbe u kojima je izgubio. Player je u *One-to-One* vezi s entitetom ban i s entitetom confirmation.

Atribut username je alternativni ključ entiteta. Atribut ban_status može poprimiti jednu od sljedećih vrijednosti: 0 - korisnik nije pod banom (nije isključen iz igre), 1 - korisnik je privremeno isključen iz igre, 2 - korisnik je trajno isključen iz igre.

player		
player_id	UUID	jedinstveni brojčani identifikator korisnika
username	VARCHAR(32)	jedinstveno korisničko ime

player		
pass_hash	VARCHAR(64)	hash lozinke
email	VARCHAR(128)	jedinstvena e-mail adresa korisnika
photo_link	VARCHAR(200)	fotografija (avatar) korisnika
points	INT	broj bodova igrača
ban_status	INT	status o zabranama igrača
activity	BOOLEAN	oznaka je li igrač online
enabled	BOOLEAN	oznaka je li igraču omogućeno korištenje računa nakon registracije
experience	INT	mjera "iskustva" u igri iskazana brojčanom vrijednošću

confirmation (Potvrda registracije) - Ovaj entitet sadržava podatke o potvrdi registracije. Sadrži atribute: ID tokena, token i ID korisnika. Ovaj entitet u vezi je *One-to-One* s Player preko jedinstvenog brojčanog identifikatora korisnika.

confirmation		
token_id	UUID	jedinstveni brojčani identifikator registracije
token	VARCHAR(255)	token potvrde o registraciji
player_id	UUID	jedinstveni brojčani identifikator korisnika

ban (Zabrana pristupa) - Ovaj entitet sadržava podatke o igračima kojima je zabranjen pristup aplikaciji. Sadrži atribute kraj zabrane i ID igrača. Ovaj entitet u vezi je *One-to-One* s korisnikom (player) preko ID-a korisnika.

ban		
player_id	UUID	jedinstveni brojčani identifikator igrača
ban_end	DATE	datum isteka zabrane

mapper (Kartograf) - Entitet mapper nasljeđuje entitet player. Ovaj entitet, uz atribute playera, ima i atribute IBAN i ID photo.

mapper		
player_id	UUID	jedinstveni brojčani identifikator kartografa
iban	VARCHAR(34)	IBAN računa za uplatu plaće
id_photo	VARCHAR(200)	fotografija osobne iskaznice

admin (Administrator) - Entitet admin nasljeđuje entitet player. Ovaj entitet ima iste atribute kao i entitet player.

admin		
player_id	UUID	jedinstveni brojčani identifikator administratora
iban	VARCHAR(34)	IBAN računa za uplatu plaće
id_photo	VARCHAR(200)	fotografija osobne iskaznice

fight (Borba) - Ovaj entitet sadržava podatke o održanim borbama između igrača. Sadrži atribute: ID borbe, trenutak početka borbe, vrijeme trajanja borbe, ID igrača koji je pobijedio i ID igrača koji je izgubio. Ovaj entitet ima dvije *Many-to-One* veze s entitetom player preko identifikatora pobjednika i gubitnika.

fight		
fight_id	UUID	jedinstveni brojčani identifikator borbe
start	TIMESTAMP	trenutak početka borbe
duration	INTERVAL	trajanje borbe
winner	UUID	jedinstveni brojčani identifikator pobjednika borbe
loser	UUID	jedinstveni brojčani identifikator gubitnika borbe

location (Lokacija) - Ovaj entitet sadržava sve važne informacije o lokacijama na kojima igrač može sakupiti karte. Sadrži atribute: ID lokacije, naziv lokacije, fotografiju lokacije i ID kategorije. Ovaj entitet u vezi je *Many-to-One* s Category preko ID kategorije te je u vezi *One-to-One* s Path preko ID lokacije.

Atribut location_status može poprimiti jednu od sljedećih vrijednosti: 0 - odobrena, 1 - odbijena, 2 - čeka odobrenje kartografa, 3 - potreban izlazak na teren i pomniji pregled

location		
location_id	UUID	jedinstveni brojčani identifikator lokacije
location_name	VARCHAR(32)	naziv lokacije
location_desc	TEXT	opis lokacije
location_photo	VARCHAR(200)	fotografija lokacije
location_status	INT	status prihvatljivosti lokacije
coordinates	VARCHAR(32)	koordinate lokacije

location		
category_id	UUID	jedinstveni brojčani identifikator kategorije

path (Najkraći put) - Ovaj entitet sadržava sve važne informacije o najkraćem putu do lokacija koje je potrebno provjeriti. Sadrži atribute. Ovaj entitet u vezi je *One-to-One* s Location preko ID lokacije.

path		
distance	INT	najkraći put do lokacije
location_id	UUID	jedinstveni brojčani identifikator lokacije

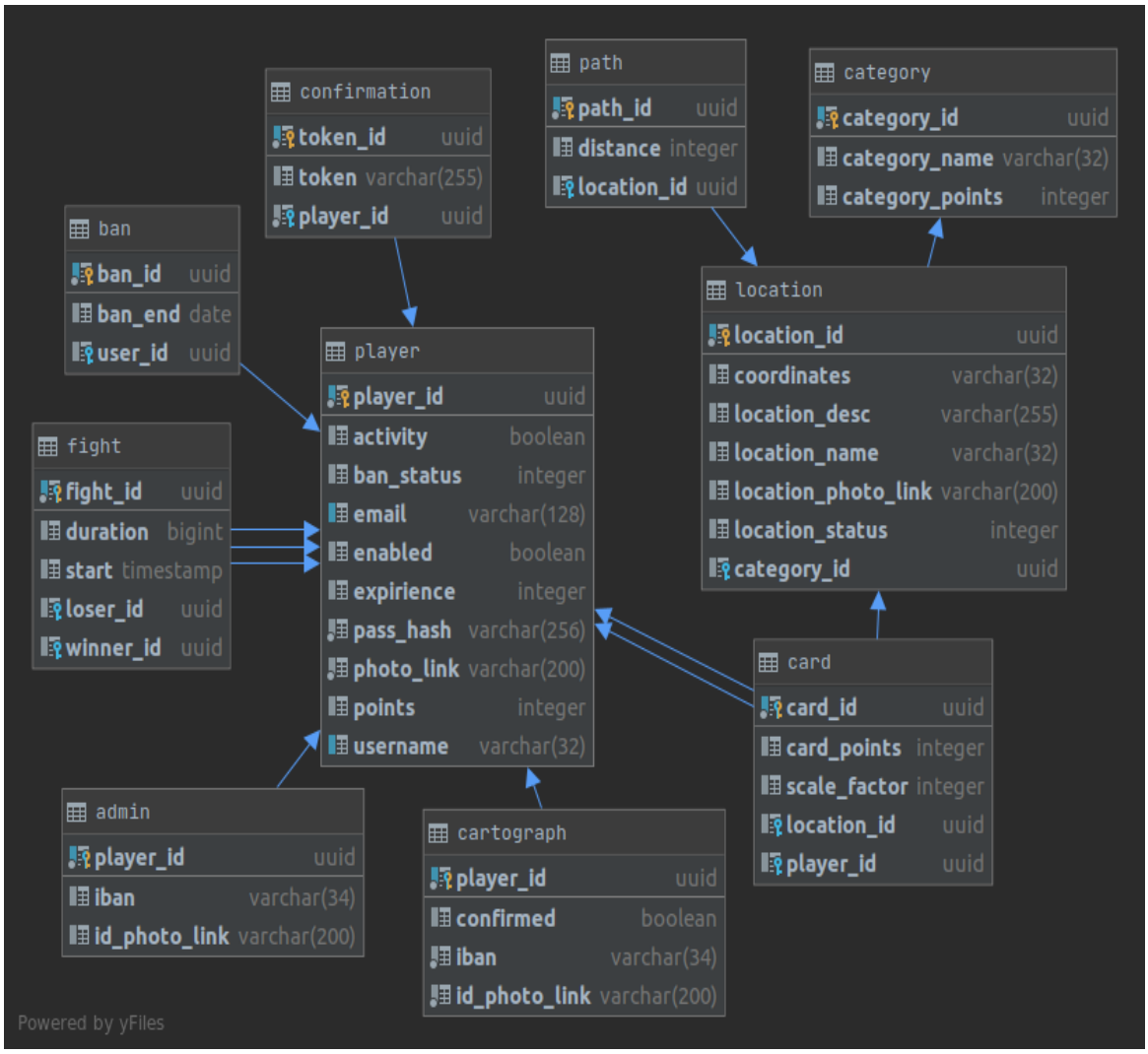
category (Kategorija lokacije) - Ovaj entitet sadržava sve važne informacije o kategorijama koje lokacije mogu biti. Sadrži atribute: ID kategorije, naziv kategorije i broj bodova koje kategorija nosi. Ovaj entitet u vezi je *One-to-Many* s Location preko ID kategorije.

category		
category_id	UUID	jedinstveni brojčani identifikator kategorije
category_name	VARCHAR(32)	naziv kategorije
category_points	INT	bodovna vrijednost kategorije

card (Karta) - Ovaj entitet sadržava sve važne informacije o kartama koje igrači mogu posjedovati. Sadrži atribute: ID karte, broj bodova karte i ID lokacije. Ovaj entitet u vezi je *Many-to-One* s Location preko ID lokacije te *Many-to-One* s korisnikom (player) preko ID-a korisnika.

card		
card_id	UUID	jedinstveni brojčani identifikator karte
card_points	INT	bodovna vrijednost karte
scale_factor	INT	faktor skaliranja bodova karte
location_id	UUID	jedinstveni brojčani identifikator lokacije
player_id	UUID	jedinstveni brojčani identifikator korisnika

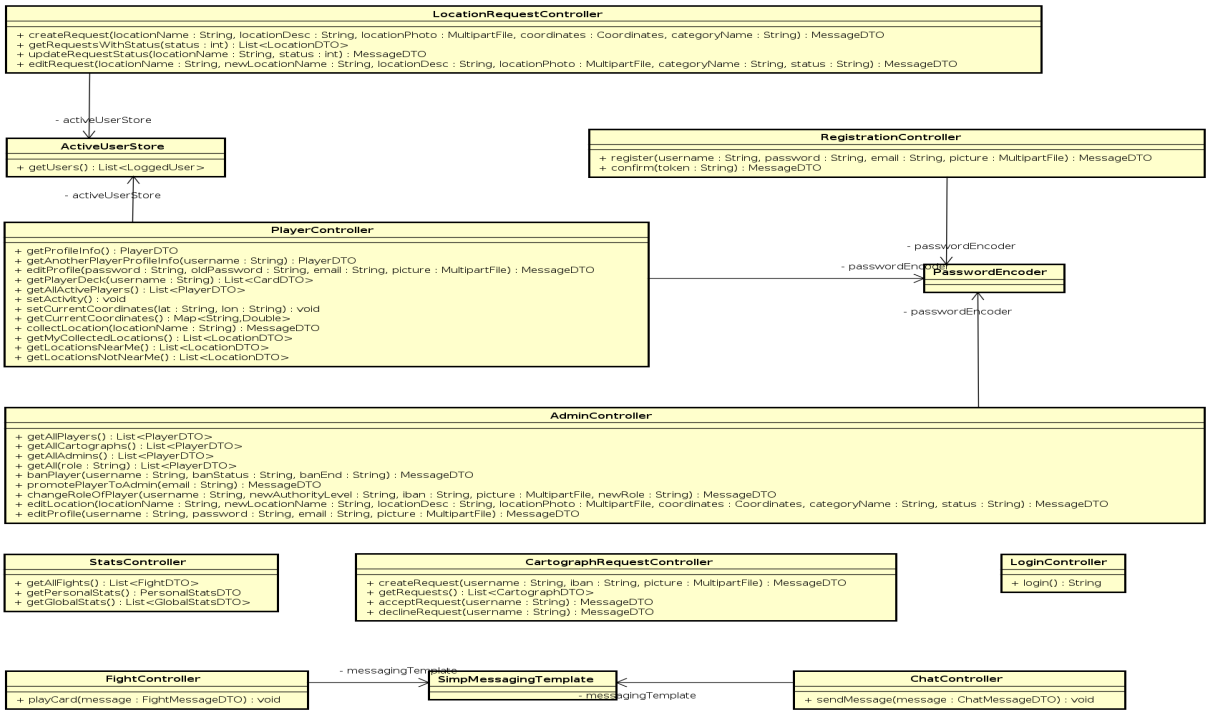
4.1.2 Dijagram baze podataka



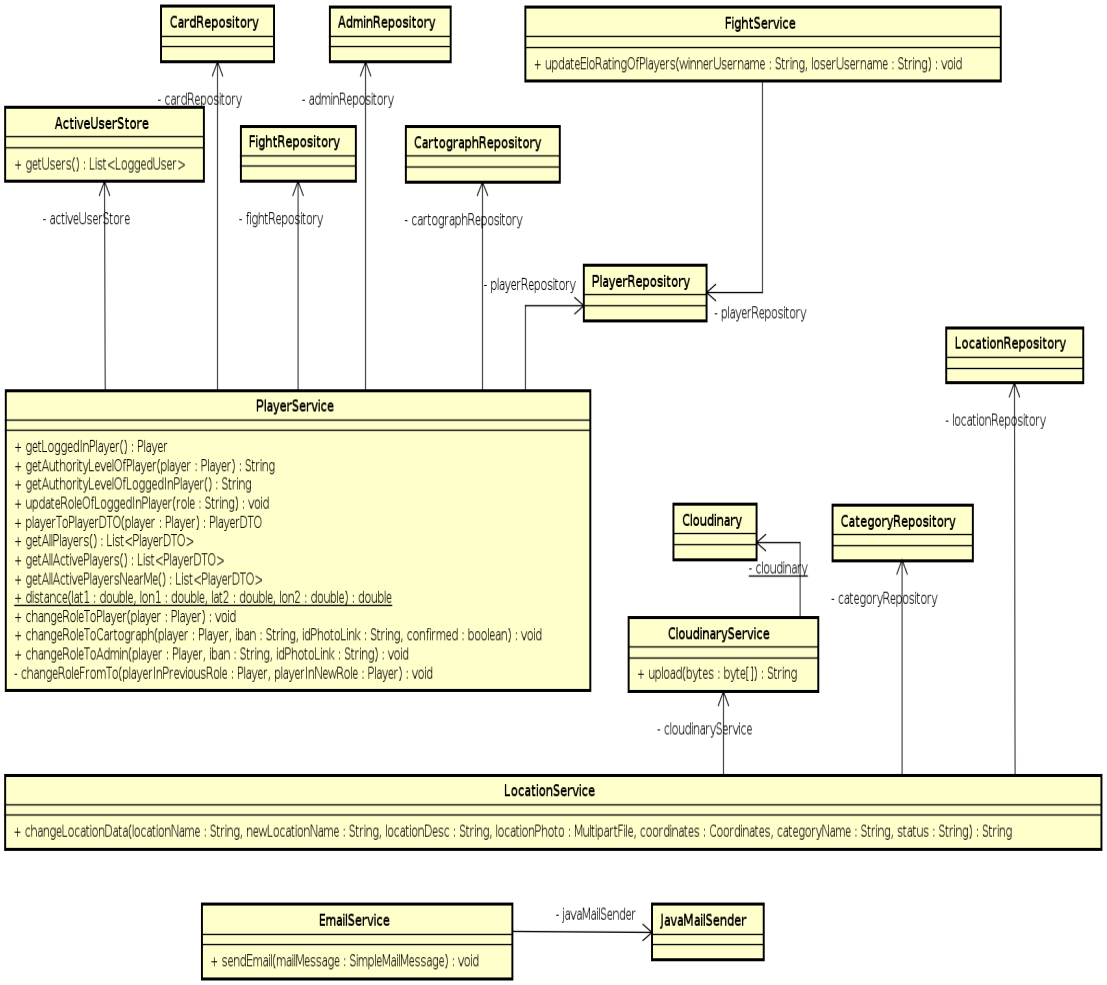
Slika 4.1: E-R dijagram baze podataka

4.2 Dijagram razreda

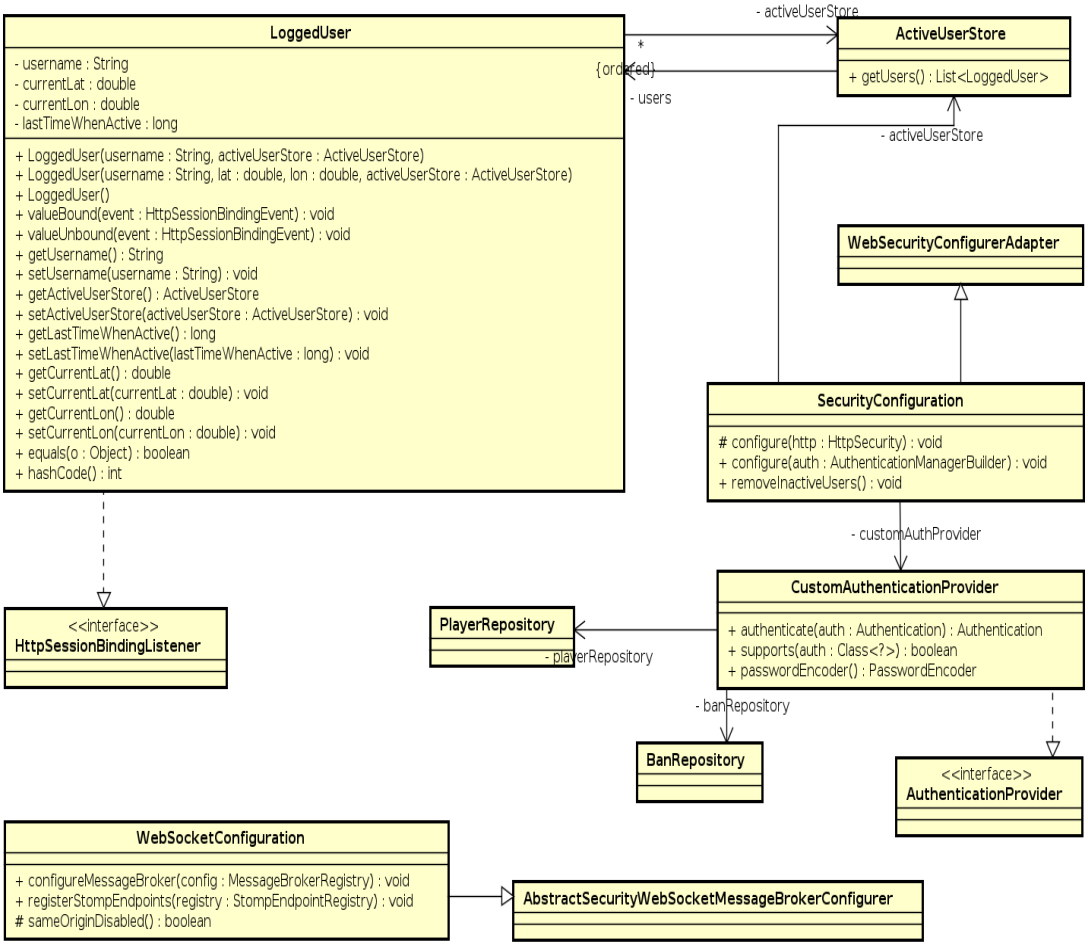
Slike 4.2, 4.3, 4.4 i 4.5 prikazuju razrede *backend* dijela MVC arhitekture. Razredi na slici 4.2 i 4.3 prikazuju razrede Service i razrede Controller s anotacijom @RestController što je specifično za spring boot koji tom anotacijom kombinira anotacije @Controller i @ResponseBody te omogućuje da svaka metoda koja rukuje sa zahtjevima automatski serijalizira povratne vrijednosti objekata u *HttpResponse*. Service razredi služe za modeliranje logike koja se događa nad modelima (npr. slanje maila) i služe za odvajanje takve logike od kontrolera čija je zadaća isključivo odgovarati na http zahtjeve (bilo GET, POST, PUT, PATCH ili DELETE).



Slika 4.2: Dijagram razreda - dio Controllers razreda

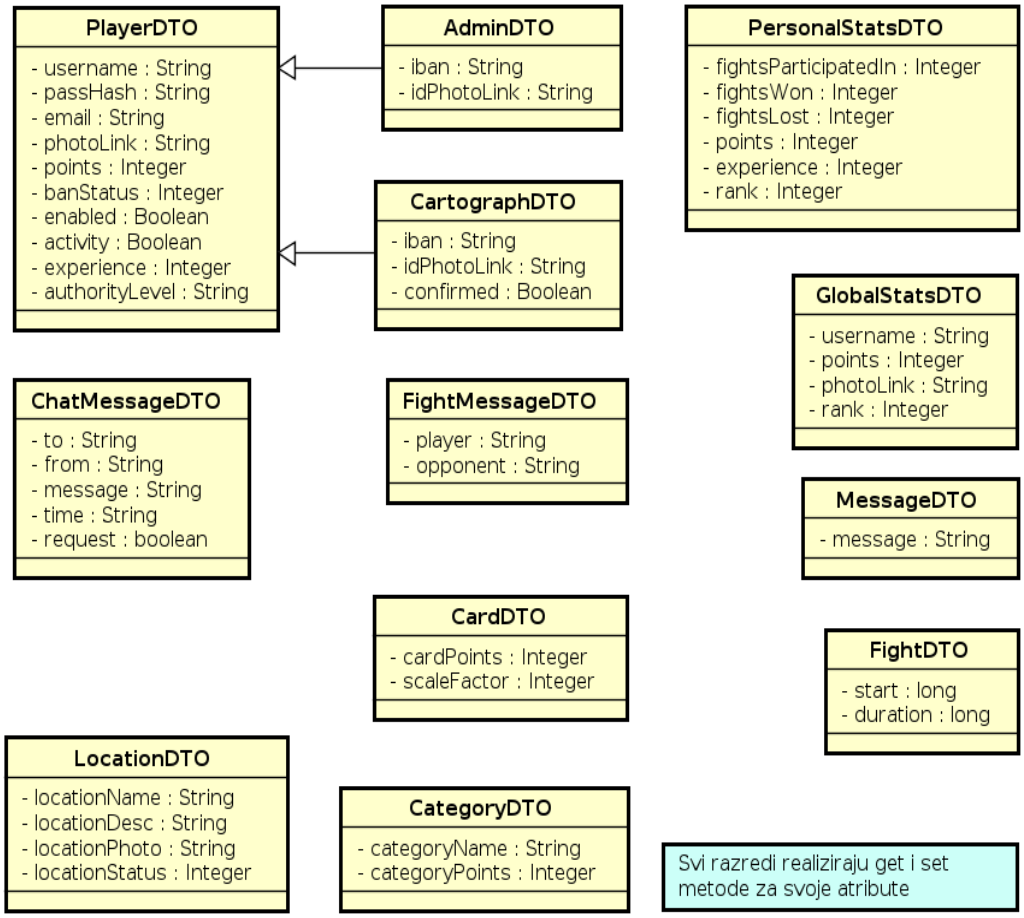


Slika 4.3: Dijagram razreda - dio Service razreda



Slika 4.4: Dijagram razreda - dio Configuration razreda

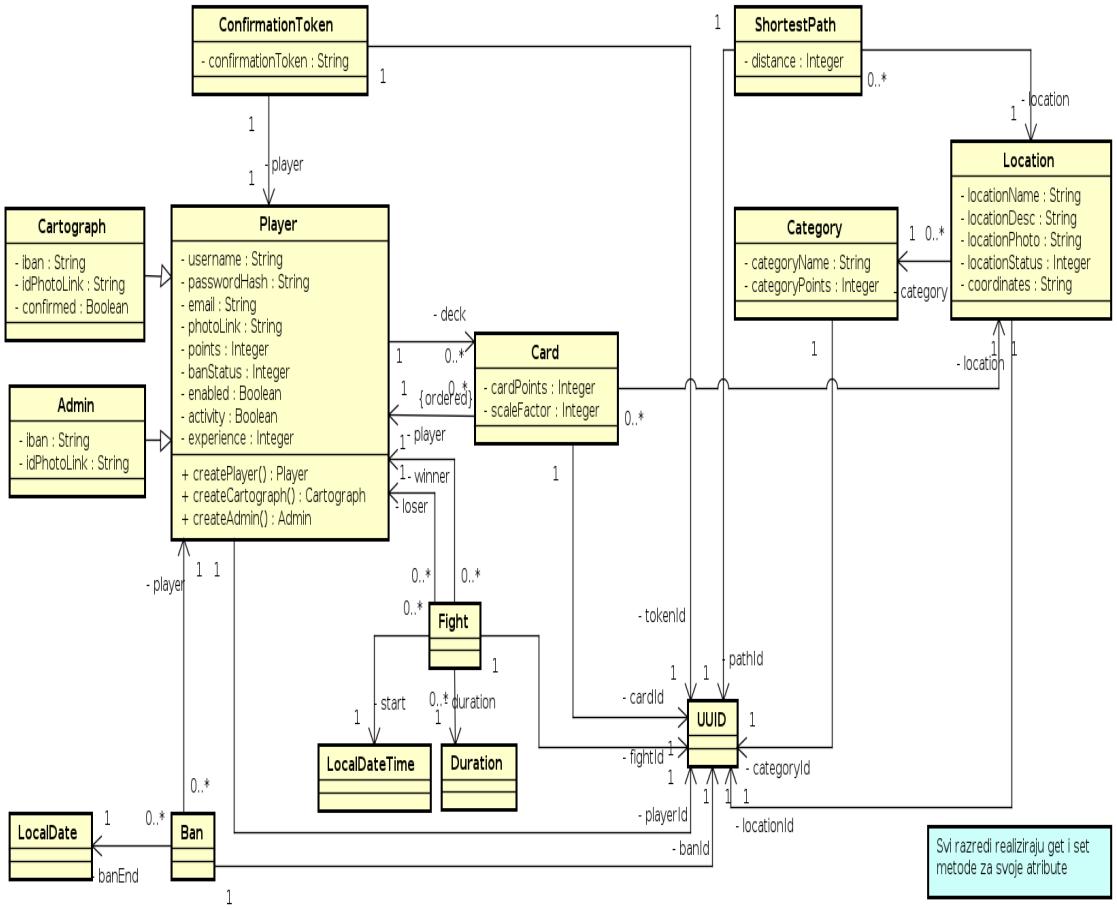
Slika 4.5: Dijagram razreda - dio Data access object razreda



Slika 4.6: Dijagram razreda - dio Data transfer object razreda

Model razredi preslikavaju strukturu baze podataka aplikacije. Razred Player predstavlja igrača koji se može registrirati, raspolaže svojim špilom karata koje skuplja te sudjeluje u borbama i otkrivanju novih lokacija. Razred Admin predstavlja administratora te nasljeđuje sve funkcije razreda Player i ima sve ovlasti nad bazom podataka i upravljanja igračima svih razina. Razred Cartograph predstavlja kartografa koji nasljeđuje sve funkcije razreda Player i ima mogućnosti upravljanja svim postojećim i novim lokacijama. Razred Fight predstavlja borbu u kojoj sudjeluju dva igrača. Razred Card predstavlja kartu koja obzirom na kategoriju lokacije kojoj pripada sadrži određen broj bodova koji se koristi u borbama. Razred Location predstavlja lokaciju na kojoj se mogu skupljati karte ukoliko ih kartograf odobri. Razred Category predstavlja kategoriju lokacije prema čijoj se klasifikaciji

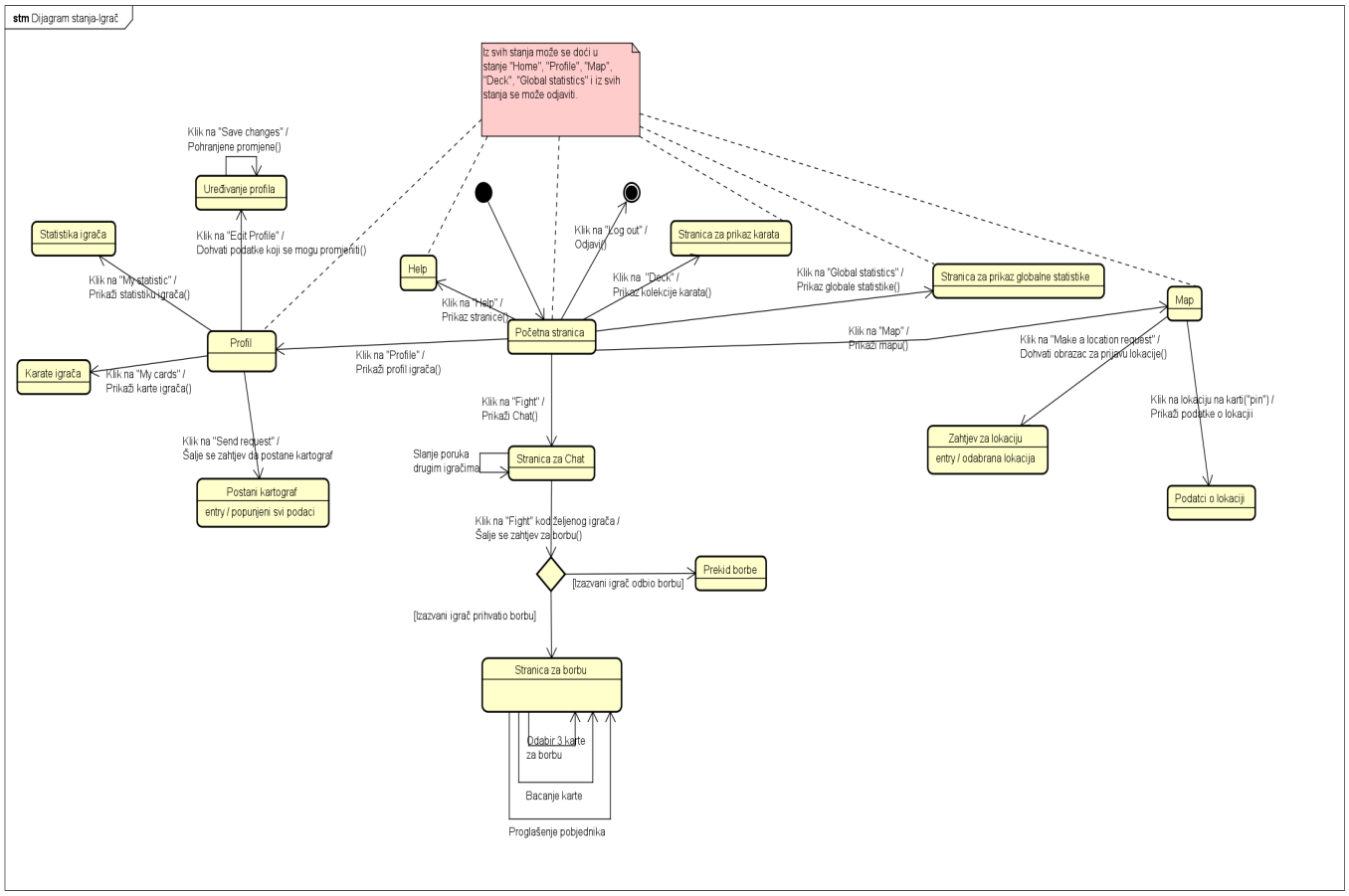
određuje broj bodova koje lokacije donose.



Slika 4.7: Dijagram razreda - dio Models razreda

4.3 Dijagram stanja

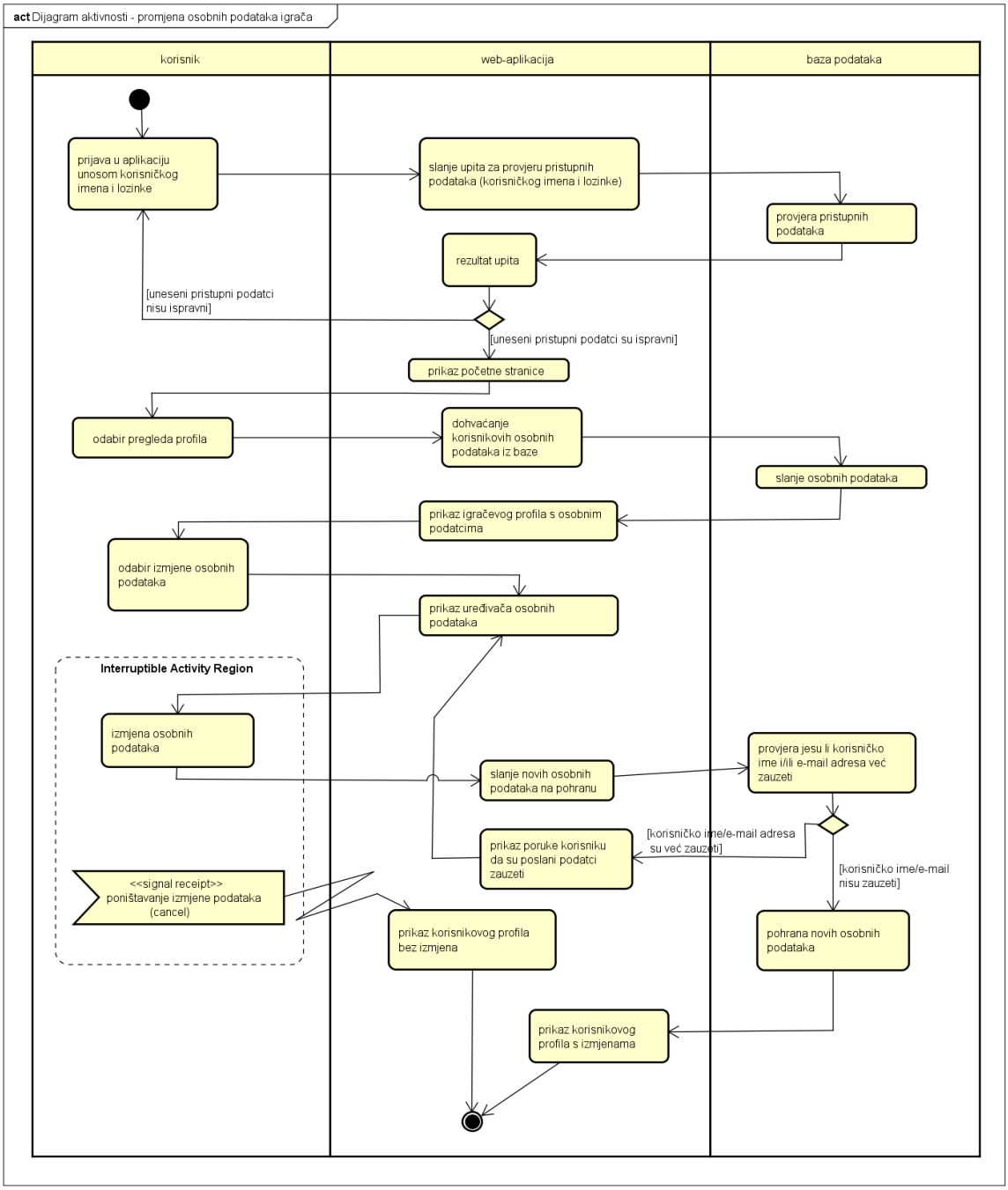
Dijagram stanja služi za opis diskretnih stanja sustava i prijelaza između tih stanja. Na slici je prikazan dijagram stanja registriranog korisnika (igrača). Nakon prijave igraču se prikazuje početna stranica (”Home”) na kojoj može prijeći na stranicu borbe. Odlaskom na tu stranicu igrač se može dopisivati s aktivnim igračima te poslati zahtjeve za borbu. Također, igrač može u padajućem izborniku odabrati stranicu za prikaz: profila (”Profile”), mape i lokacija (”Map”), kolekciju karata (”Deck”), globalne statistike (”Global statistics”), te stranica ”Help”. U padajućem izborniku može se i vratiti na početnu stranicu te iz bilo koje stranice se može odjaviti. Na stranici profila igrač može mijenjati osobne podatke, pregledati vlastitu statistiku i kolekciju karata. Također može ispuniti i poslati zahtjev da postane kartograf. Na stranici ”Map” igrač vidi dostupne lokacije (i njihove karakteristike) te može predati zahtjev za novu lokaciju. Na stranici za prikaz karata igrač vidi kolekciju karta, a na stranici globalne statistike može vidjeti svoj rang na globalnoj statistici.



Slika 4.8: Dijagram stanja

4.4 Dijagram aktivnosti

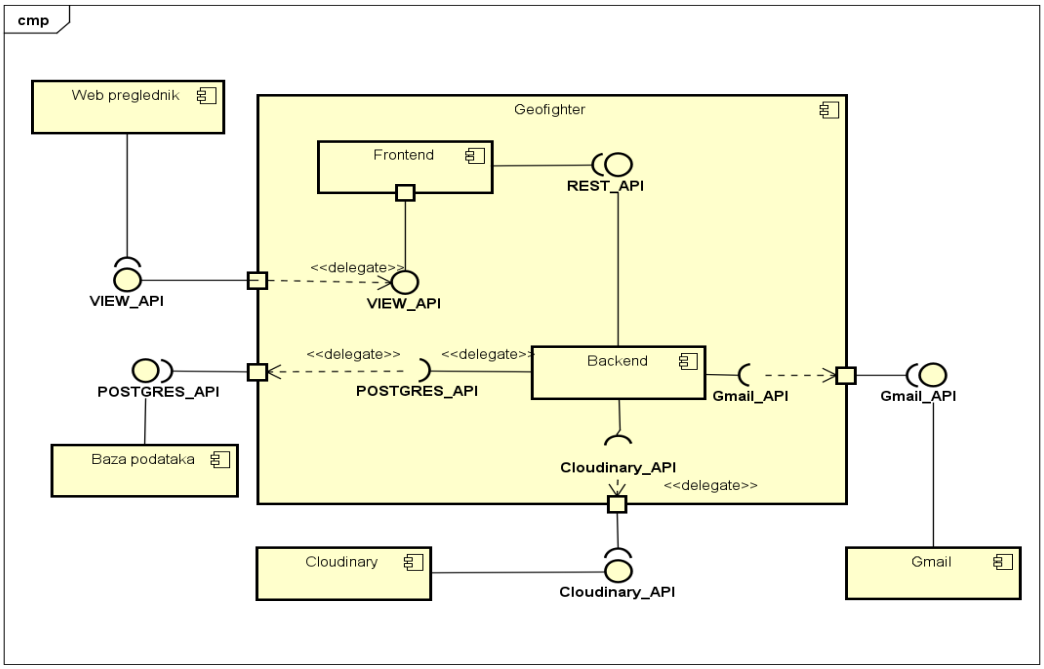
UML dijagram aktivnosti je dijagram koji prikazuje neku aktivnost kao jednu cjelinu koja se sastoji od niza akcija. Dijagram aktivnosti može se primjenjivati za modeliranje nečega na visokoj razini apstrakcije kao što su poslovni procesi, ili za modeliranje na niskoj razini apstrakcije kao što je oblikovanje detalja nekog algoritma. Dijagrami aktivnosti ne primjenjuju se za modeliranje događajima poticajnog ponašanja, jer se kod dijagrama aktivnosti (za razliku od dijagrama stanja) podrazumijeva da jedna akcija slijedi drugu. Slika 4.9 prikazuje dijagram aktivnosti kojim se modelira tijek akcija kod promjene osobnih podataka korisnika. Korisnik se prvo prijavljuje u aplikaciju korisničkim imenom i lozinkom, zatim s početne stranice odabire pregled vlastitog profila. Na profilu postoji gumb koji otvara uređivač osobnih podataka. Korisnik u uređivaču može promijeniti svoje osobne podatke i spremiti izmjene ili odustati od izmjena pritiskom na jedan od dva gumba. Ako korisnik unese podatak kao što je e-mail koji neki drugi korisnik već koristi i pokuša pohraniti promjenu, aplikacija će izbaciti poruku da se unesena e-mail adresa već koristi. Nakon zatvaranja uređivača korisnik ponovno vidi svoju stranicu profila s ažuriranim podatcima (ili starim podatcima ukoliko je korisnik odustao od izmjena).



Slika 4.9: Dijagram aktivnosti

4.5 Dijagram komponenti

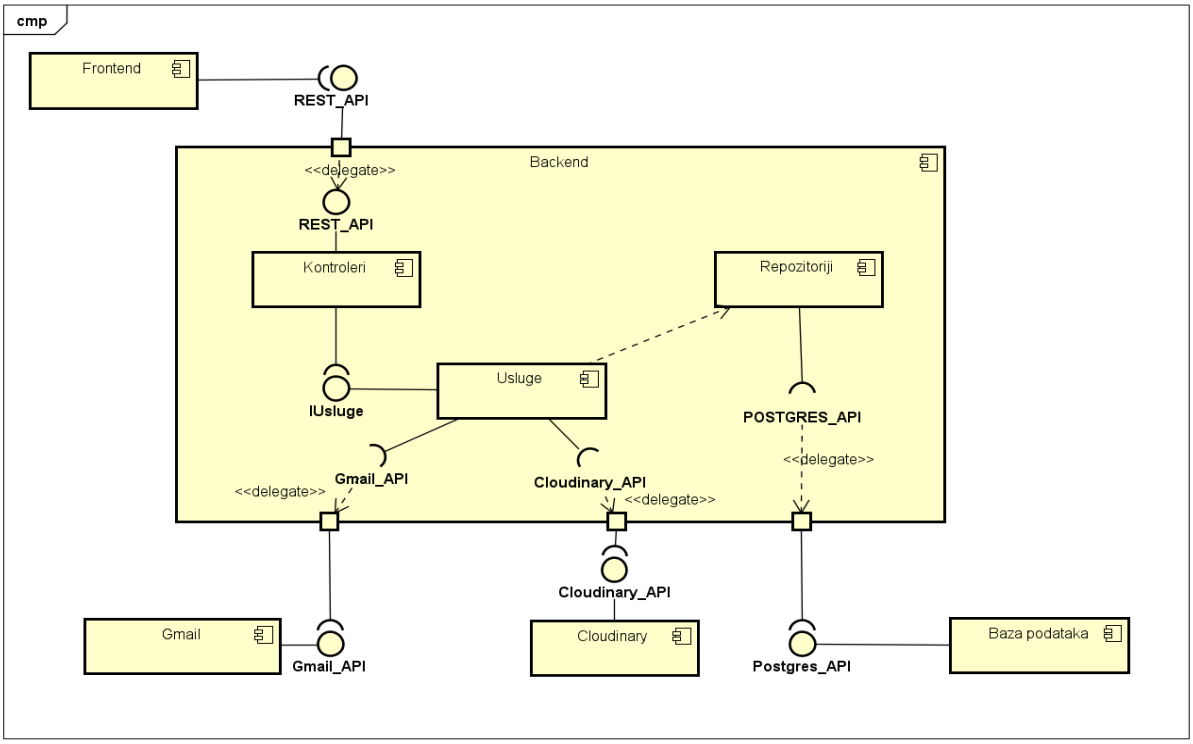
Dijagram komponenti opisuje arhitekturu programske potpore tako da vizualizira organizaciju i međuovisnost između implementacijskih komponenti te odnos programske potpore prema okolini. Unutar naše osnovne komponente Geofighter koja opisuje samu aplikaciju stvorili smo dvije glavne podkomponente - Backend i Frontend. Geofighter komunicira s komponentom Baza podataka koja nudi mogućnost spremanja i dohvata podataka preko vanjskog sučelja POSTGRES_API. Ono unutar Geofighter-a delegira istoimeno unutarnje sučelje koje zahtijeva komponenta Backend. Backend, osim s Bazom podatka, zahtijeva i sučelja Cloudinary_API i Gmail_API koja delegiraju također istoimena vanjska sučelja. Vanjsko sučelje Cloudinary_API pruža funkcionalnost pohrane slika te njega nudi komponenta Cloudinary. Komponenta Gmail nudi sučelje Gmail_API pomoću kojega Geofighter šalje e-mail potvrde prilikom registracije igrača. S druge strane, Geofighter mora komunicirati i s komponentom Web preglednik, a to ostvaruje preko Frontenda i vanjskog i unutarnjeg sučelja VIEW_API kojega nudi Frontend i pomoću kojega se korisniku prikazuje aplikacija.



Slika 4.10: Dijagram komponenti - Geofighter

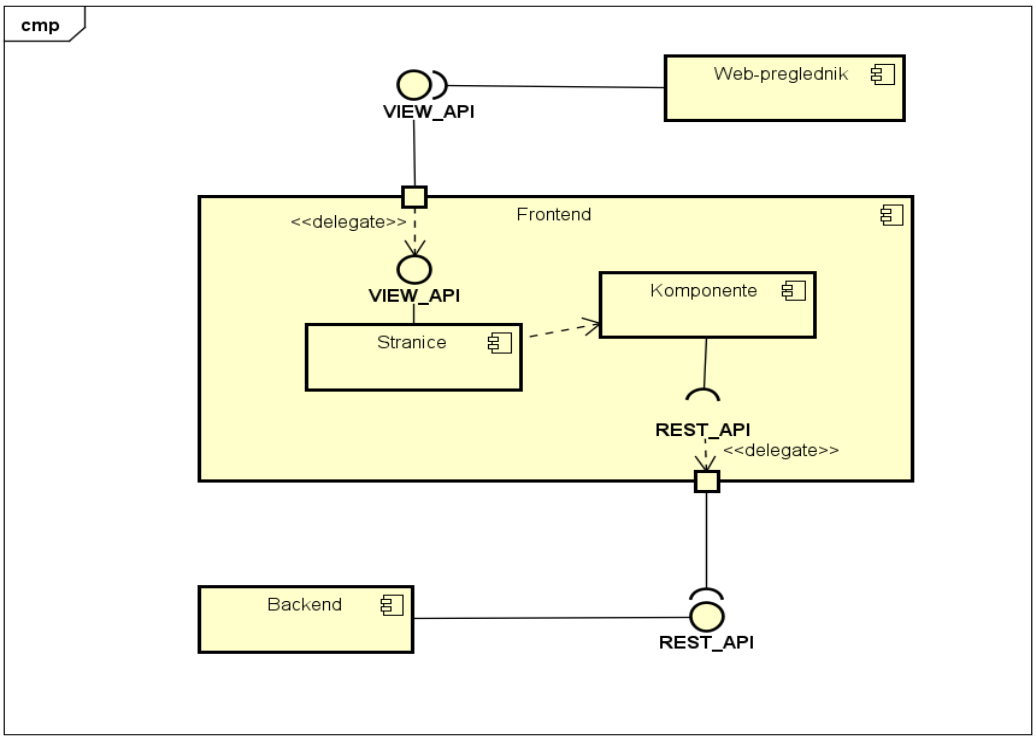
Unutar Backenda nalaze se tri podkomponente: Repozitoriji, Usluge i Kontroleri. Repozitoriji su povezani preko unutarnjeg sučelja delegiranog iz POS-

TGRES_API sučelja te oni služe za upravljanje podatcima iz baze podataka koje onda pružaju uslugama kada je to potrebno. Usluge su komponenta preko koje Backend komunicira s komponentama Cloudinary i Gmail preko već navedenih sučelja. Usluge komponenti Kontroleri nude sučelje IUsluge preko kojega se obavljaju određene funkcionalnosti u interakciji s Frontend komponentom. Kontroleri zatim podatke i usluge delegiraju komponenti Frontend preko sučelja REST_API.



Slika 4.11: Dijagram komponenti - Backend

Komponenta Frontend ima dvije unutarnje komponente, a to su Komponente i Stranice. Komponente zahtijevaju sučelje REST_API pomoću kojega primaju podatke preko http metoda koje zatim koriste u vizualizaciji određenih dijelova aplikacije za koje su stvorene. Stranice se sastoje od jedne ili više komponenti koje objedinjuju u cjeline i zatim prikazuju Web-pregledniku preko ponuđenog sučelja VIEW_API.



Slika 4.12: Dijagram komponenti - Frontend

5. Implementacija i korisničko sučelje

5.1 Korištene tehnologije i alati

Komunikacija unutar grupe najviše se odvijala preko aplikacije Discord[1], a manjim dijelom u početku putem Microsoftove platforme za poslovnu komunikaciju Teams[2]. Za izradu gotovo svih UML dijagrama korišten je alat Astah UML[3], dok je dijagram baze podataka napravljen uz pomoć razvojnog okruženja DataGrip[4] tvrtke JetBrains. Upravljanje različitim verzijama koda i dokumentacije obavljanno je uz pomoć sustava Git[5], a podatci su se čuvali u udaljenom repozitoriju na platformi GitLab[6].

Članovi tima koji su izrađivali *backend* koristili su IntelliJ IDEA[7] ili Eclipse[8] kao razvojno okruženje, dok su članovi na *frontendu* koristili *editor* Visual Studio Code[9]. IntelliJ i Eclipse su razvojna okruženja primarno namijenjena za razvoj softvera u Javi; Eclipseova funkcionalnost može se znatno proširiti *plug-inovima*. Visual Studio Code jednostavan je uređivač izvornog koda koji se može koristiti s raznim programskim jezicima, a razvio ga je Microsoft.

Za izradu *backenda* aplikacije korišteni su radni okvir Spring Boot[10] i jezik Java[11]. Spring Boot je specijalizacija radnog okvira Spring[12] (ne nadomješta Spring) koja pojednostavljuje oblikovanje web aplikacije jer automatski konfigurira važne funkcionalnosti Springa. Na *frontend* strani aplikacije korišteni su React[13] i JavaScript[14]. React tehnički nije radni okvir, nego knjižnica pisana u JavaScriptu, iako se smatra

[1] https://discord.com/
[2] https://www.microsoft.com/hr-hr/microsoft-teams/
[3] https://astah.net/products/astah-uml/
[4] https://www.jetbrains.com/datagrip/
[5] https://git-scm.com/
[6] https://gitlab.com/
[7] https://www.jetbrains.com/idea/
[8] https://www.eclipse.org/
[9] https://code.visualstudio.com/
[10] https://spring.io/projects/spring-boot
[11] https://www.java.com/
[12] https://spring.io/projects/spring-framework
[13] https://reactjs.org/
[14] https://www.javascript.com/

jednim od najznačajnijih *frontend* radnih okvira u vrijeme izrade ovog projekta.

Za neke funkcionalnosti aplikacije korišteni su vanjski servisi: OpenStreetMap[15] za prikaz karte na kojoj se vide lokacije, OSRM[16] za računanje kartografovog puta do lokacije koju treba provjeriti i Cloudinary[17] za *hostanje* slika lokacija i profilnih slika korisnika.

Aplikacija je puštena u pogon na poslužitelju Heroku[18], a baza podataka povezana je na Heroku preko Amazon AWS[19] usluge.

[15] https://www.openstreetmap.org
[16] http://project-osrm.org/
[17] https://cloudinary.com/
[18] https://www.heroku.com/
[19] https://aws.amazon.com/products/databases/

5.2 Ispitivanje programskog rješenja

5.2.1 Ispitivanje komponenti

Ispitivanje komponenti provodimo pomoću JUnit javinog okvira za testiranje i pisanjem ispitnih slučajeva korištenjem jednog od javinih IDE, Eclipse ili IntelliJ.

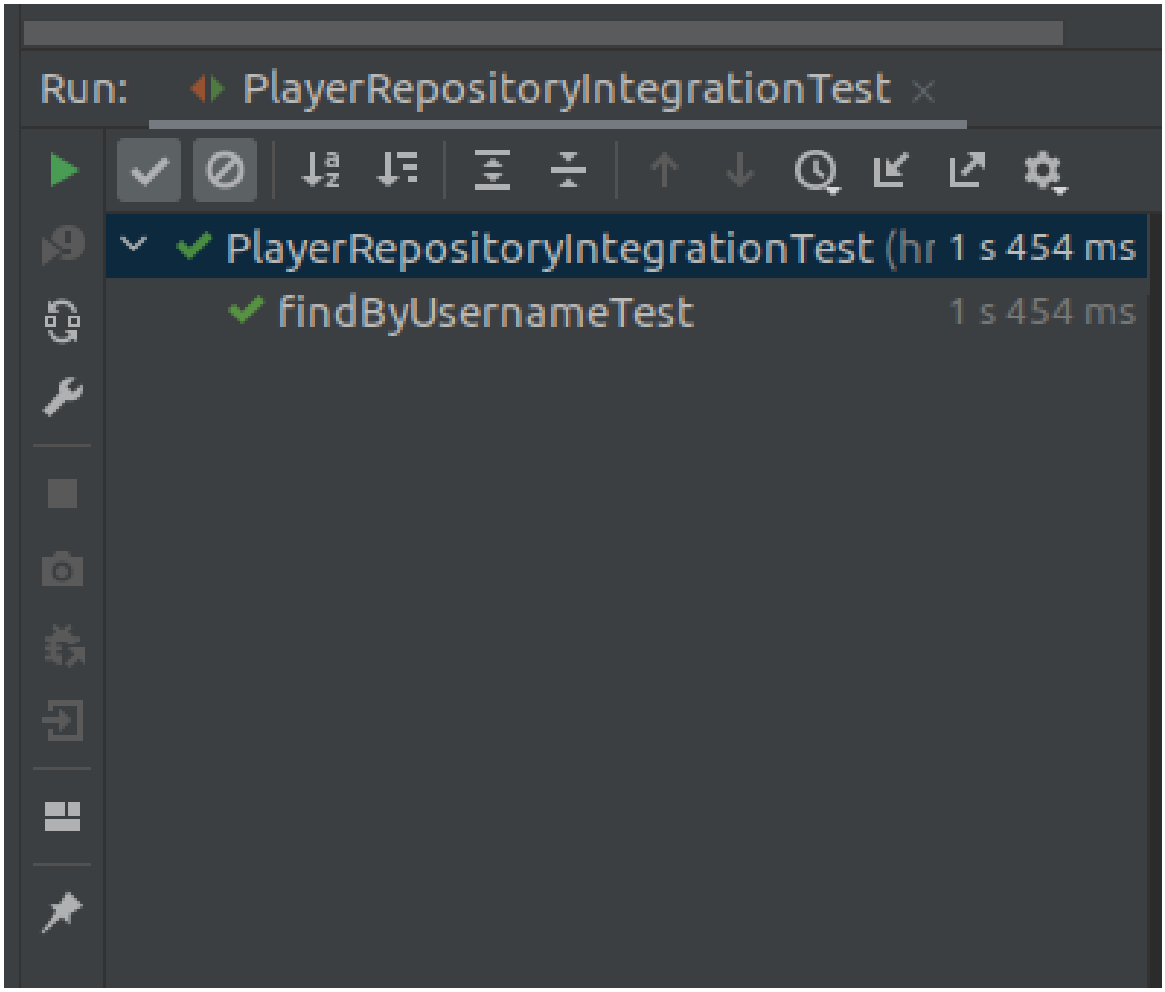
1. Ispitivanje traženja igrača korisničkim imenom

Prilikom pisanja programskog koda za ovaj test koristi se anotacija *@DataJpaTest* koji je specifičan za radni okvir *spring* prilikom rada s *Data Transfer Object* razredima. Očekivamo da će test biti uspješan i uspjeti pronaći igrača u bazi.

```
@Test
public void findByUsernameTest() {
    Player igrac5 = new Player("igrac5", "igrac", "igrac5@fer.hr", "");
    entityManager.persist(igrac5);
    entityManager.flush();

    Player player = playerRepository.findByUsername(igrac5.getUsername());

    assertEquals(igrac5.getUsername(), player.getUsername());
}
```

Slika 5.1: Player Repository - JUnit, test

Rezultat



Slika 5.2: Player Repository - JUnit, rezultat

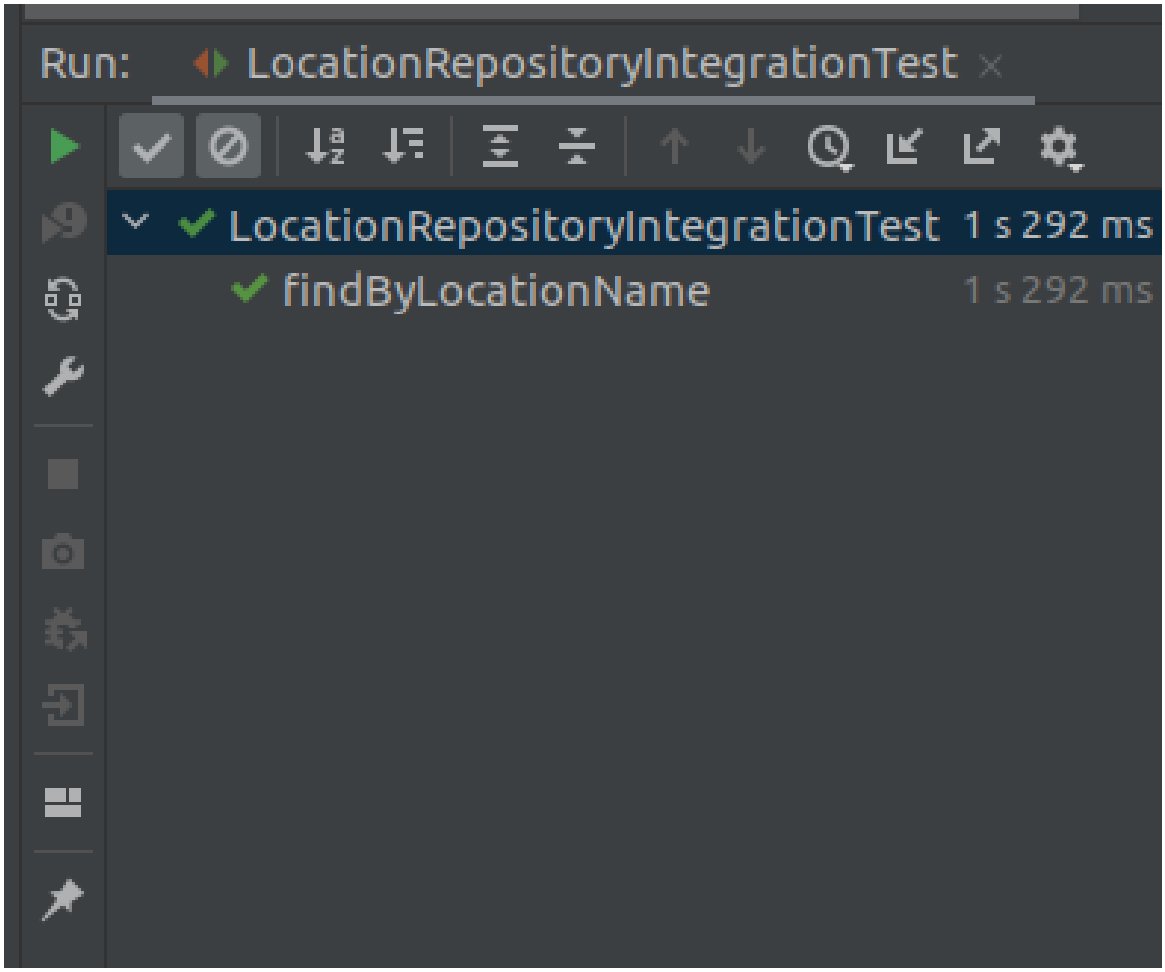
2. Ispitivanje traženja lokacije imenom

Očekivamo da će test biti uspješan i uspjeti pronaći lokaciju u bazi.

```
@Test
public void findByLocationName() {
    Location cakovec = new Location( name: "Cakovec",  desc: "",  photo: "",  coordinates: "", categoryRepository.findByCategoryName("Grad"));
    entityManager.persist(cakovec);
    entityManager.flush();

    Location location = locationRepository.findByLocationName(cakovec.getLocationName());

    assertEquals(cakovec.getLocationName(), location.getLocationName());
}
```

Slika 5.3: Location Repository - JUnit, test

Rezultat



Slika 5.4: Location Repository - JUnit, rezultat

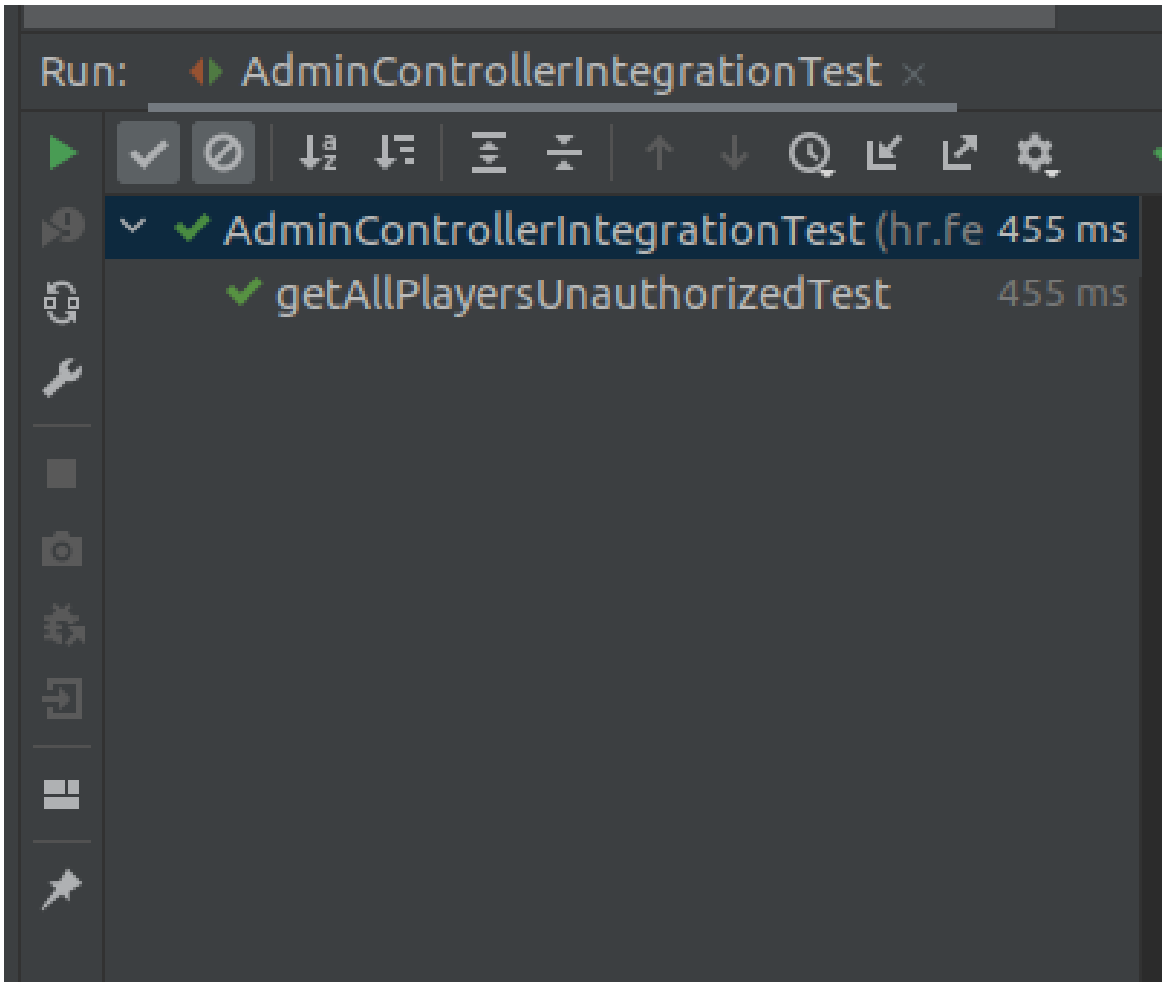
3. Ispitivanje traženja bespravnog zahtjeva za popis svih igrača

Prilikom pisanja ovih testova koristi se anotacija *@SpringBootTest* i *@AutoConfigureMockMvc*. Očekivamo da će test biti uspješan jer se tijekom pretrage ne može obaviti provjera nad prijavljenim korisnikom kako bi se ustvrdilo da ima razinu adminstratorovih ovlasti.

```
@Test
public void getAllPlayersUnauthorizedTest() throws Exception {
    mvc.perform(MockMvcRequestBuilders.get( urlTemplate: "/allPlayers")
            .accept(MediaType.APPLICATION_JSON))
            .andDo(print())
            .andExpect(status().isUnauthorized());
}
```

Slika 5.5: Admin Controller - JUnit, test

Rezultat



Slika 5.6: Admin Controller - JUnit, rezultat

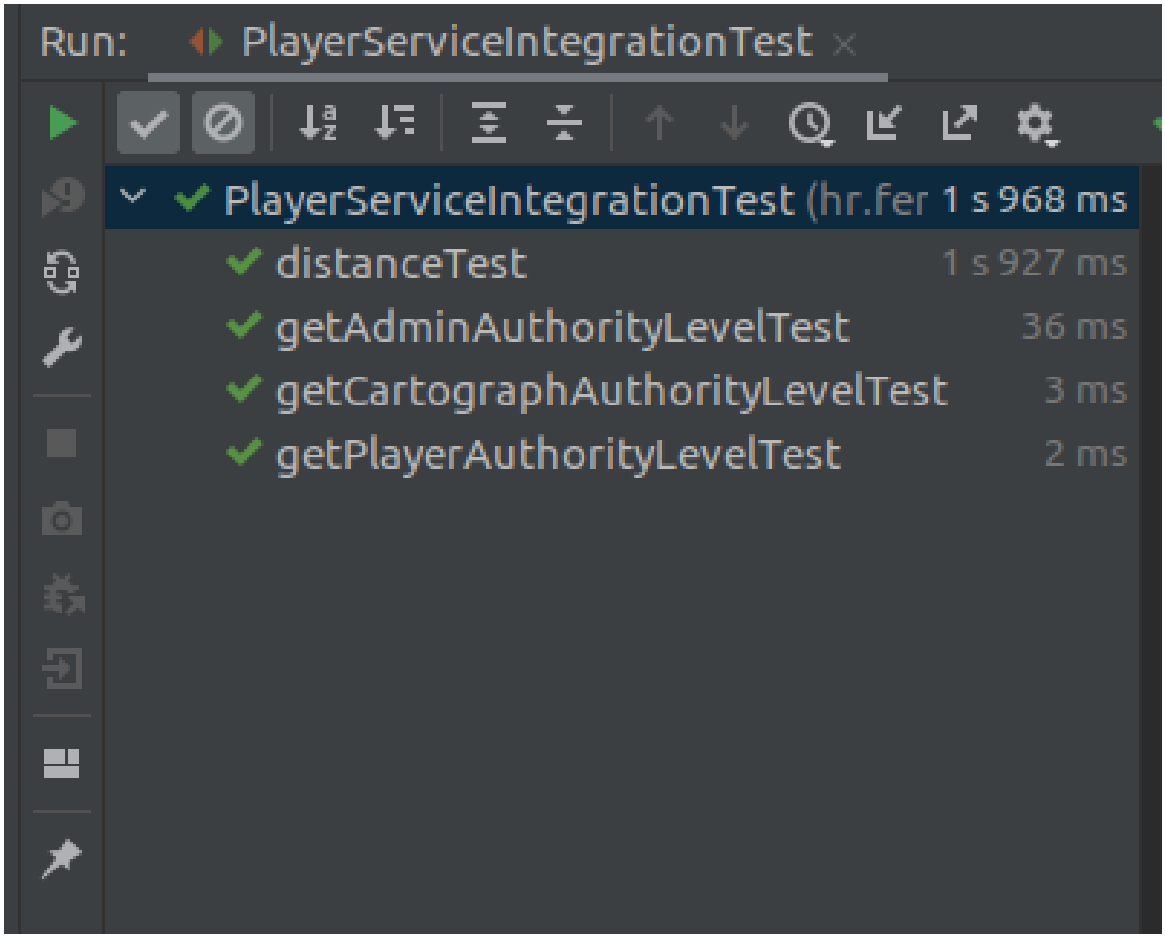
4. Ispitivanje razina ovlasti igrača i računanja udaljenosti

Prilikom pisanja ovih testova koristi se anotacija *@Mock* specifična za rad sa Service razredima. U ovom primjeru testiramo više metoda u razredu Player Service. Ispitujemo razine ovlasti kod postojećih igrača te metodu za računanje udaljenosti. Očekujemo da će za dane ulaze testovi proći.

```
@Test
public void getPlayerAuthorityLevelTest() {
    Player player = playerRepository.findByUsername("igrac1");
    Mockito.when(playerService.getAuthorityLevelOfPlayer(player)).thenReturn("player");

    assertEquals( expected: "player", playerService.getAuthorityLevelOfPlayer(player));
}

@Test
public void getCartographAuthorityLevelTest() {
    Cartograph cartograph = cartographRepository.findByUsername("kartograf2");
    Mockito.when(playerService.getAuthorityLevelOfPlayer(cartograph)).thenReturn("player");

    assertEquals( expected: "player", playerService.getAuthorityLevelOfPlayer(cartograph));
}

@Test
public void getAdminAuthorityLevelTest() {
    Admin admin = adminRepository.findByUsername("admin1");
    Mockito.when(playerService.getAuthorityLevelOfPlayer(admin)).thenReturn("admin");

    assertEquals( expected: "admin", playerService.getAuthorityLevelOfPlayer(admin));
}

@Test
public void distanceTest() {
    assertEquals( expected: 5503.5539, playerService.distance( lat1: 30,  lon1: 30,  lat2: 60,  lon2: 90),  delta: 0.001);
}
```

Slika 5.7: Player Service - JUnit, test

Rezultat



Slika 5.8: Player Service - JUnit, rezultat

5.2.2 Ispitivanje sustava

Ispitivanje sustava možemo provesti pomoću dodatka za preglednik Selenimu IDE i pisanjem ispitnih slučajeva korištenjem programskog sučelja, tj. mi koristimo Selenium WebDriver unutar JUnit testova. Pomoću dodatka za preglednik Selenium IDE snimaju se korisnikove akcije radi automatskog ponavljanja ispitnog slučaja.

1. Ispitivanje dobre prijave na web aplikaciju GeoFighter

Na ulazu unosimo podatke za prijavu već registriranog igrača (*username: igrac1* i *password: igrac*). Očekivani izlaz je preusmjeravanje na stranicu */home*. Preusmjeravanje pratimo prema elementima s određenih stranica.

```
@Test
public void testLoginGoodCreds() {
    WebDriver driver = new ChromeDriver();
    System.setProperty("webdriver.chrome.driver", "C:\\Program Files (x86)\\Chrome Driver\\chromedriver.exe");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    driver.get("https://ferllowship-testing.herokuapp.com/");


    WebElement element = driver.findElement(By.cssSelector("input[placeholder='Username']"));
    element.sendKeys("igrac1");

    element = driver.findElement(By.cssSelector("input[placeholder='Password']"));
    element.sendKeys("igrac");

    driver.findElement(By.className("btn")).click();

    boolean find= driver.findElement(By.className("btnFight")) != null;

    assertEquals(find, true);

    driver.quit();

}
```

Slika 5.9: Selenium WebDriver - JUnit, test1

2. Ispitivanje loše prijave na web aplikaciju GeoFighter

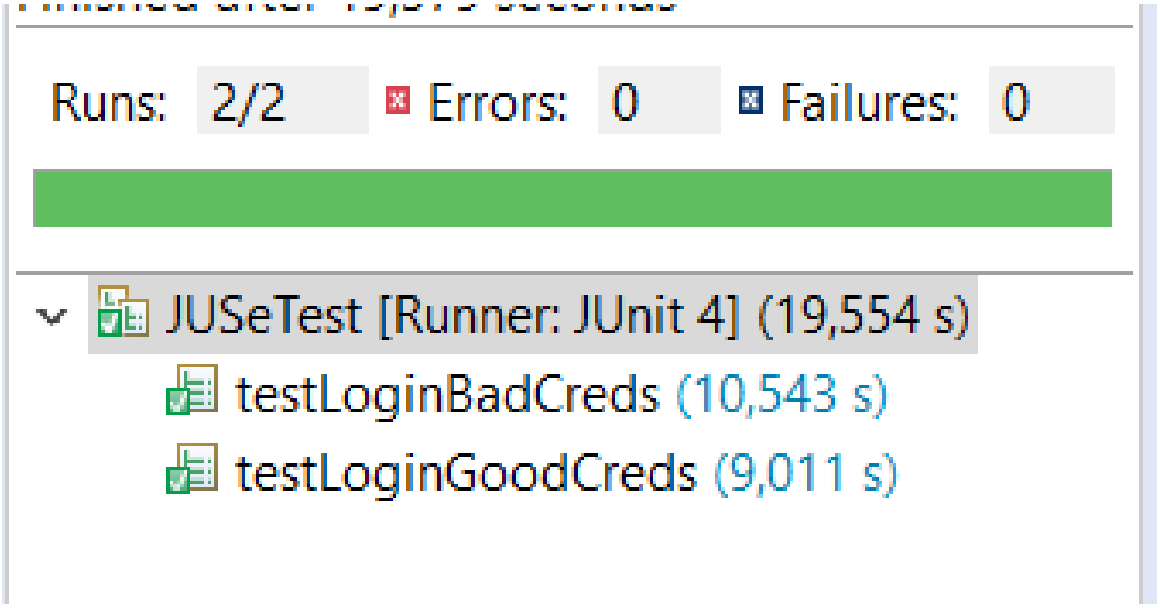
Na ulazu unosimo podatke za neregistriranog igrača (*username: testadmin* i *password: 22345*). Očekivani izlaz je ostanak na stranici za prijavu, što provjeravamo pomoću karakterističnih elemenata sa stranice.

```
@Test
public void testLoginBadCreds() {

    System.setProperty("webdriver.chrome.driver", "C:\\Program Files (x86)\\Chrome Driver\\chromedriver.exe");
    WebDriver driver = new ChromeDriver();
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    driver.get("https://ferllowship-testing.herokuapp.com/");


    WebElement element = driver.findElement(By.cssSelector("input[placeholder='Username']"));
    element.sendKeys("testadmin");

    element = driver.findElement(By.cssSelector("input[placeholder='Password']"));
    element.sendKeys("22345");

    driver.findElement(By.className("btn")).click();

    boolean find= driver.findElement(By.className("btn")) != null;

    assertEquals(find, true);

    driver.quit();

}
```

Slika 5.10: Selenium WebDriver - JUnit, test2

Rezultat

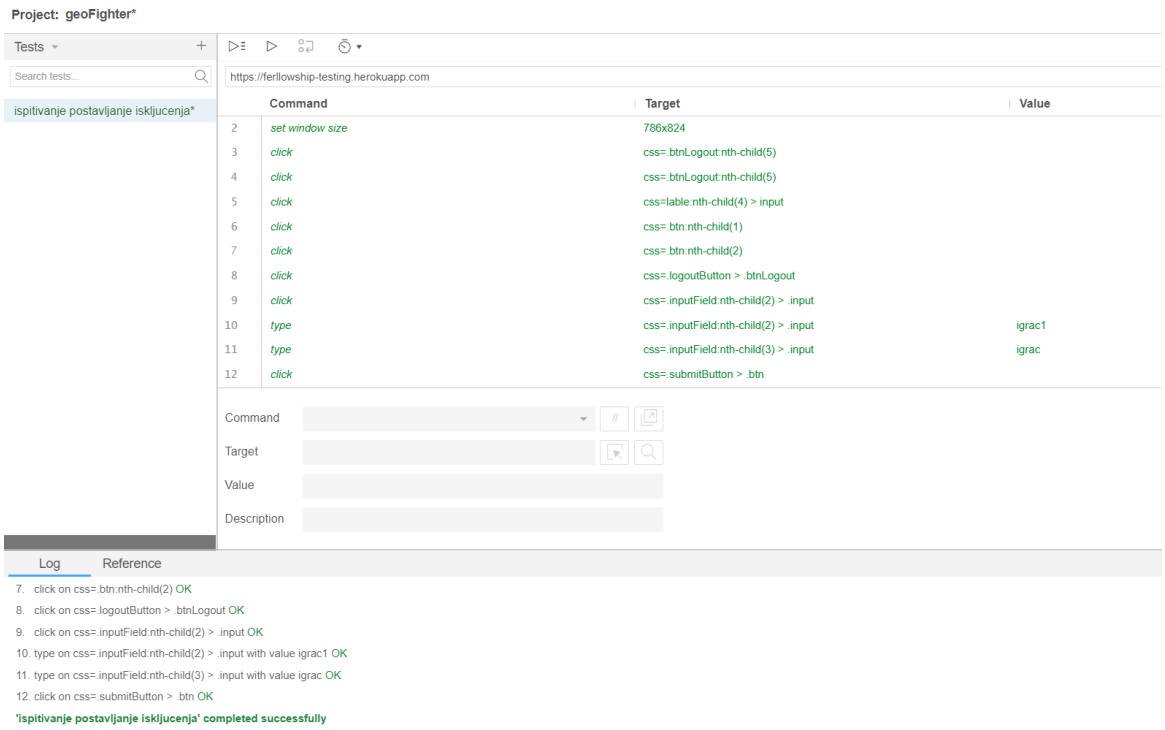
Nakon pokretanja testova *testLoginGoodCreds* i *testLoginBadCreds*, dobili smo očekivani rezultat. Odnosno aplikacija je **prošla** testove.



Slika 5.11: Selenium WebDriver - JUnit, rezultat

3. Ispitivanje postavljanja isključenja

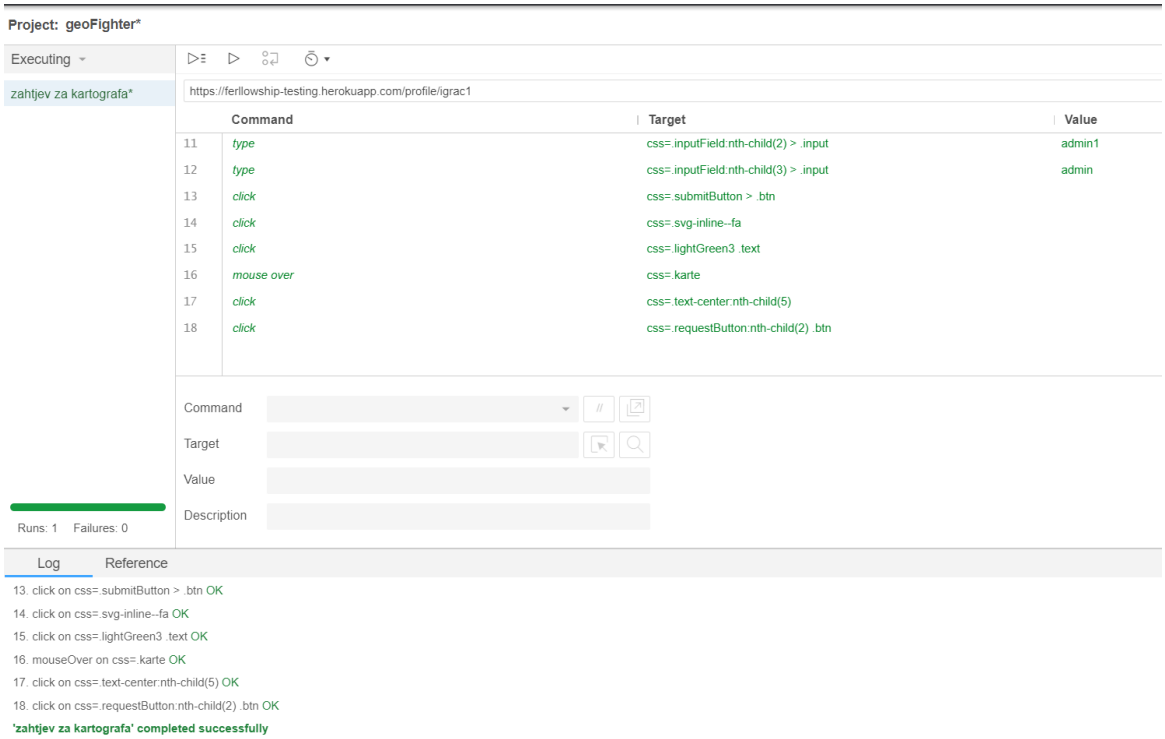
Ovo ispitvanje provodimo pomoću dodatka na Google Chrome *Selenium IDE*. Prijavimo se kao *admin1*, pregledamo profil *igraca1*. Na ulazu testa unosimo trajno isključenje za *igraca1* te se odjavimo kao *admin1*. Očekivani izlaz je kad se pokušamo prijaviti kao *igrac1* da nam to neće biti dozvoljeno dok će nam prijava kao *igrac2* biti dozvoljena. Te radnje smo snimili pomoću Selenium IDE dodatka te ih pokrenuli s podatcima za *igrac1*. Ako ponovno pokrenemo test i promijenimo korisničko ime na *igrac2* vidimo da test dobro radi, odnosno s *igrac2* smo se uspjeli prijaviti. Dobili smo željeni rezultat. Aplikacija je **prošla** test.



Slika 5.12: Selenium IDE, ispitivanje postavljanja isključenja

4. Ispitivanje zahtjeva za kartografa

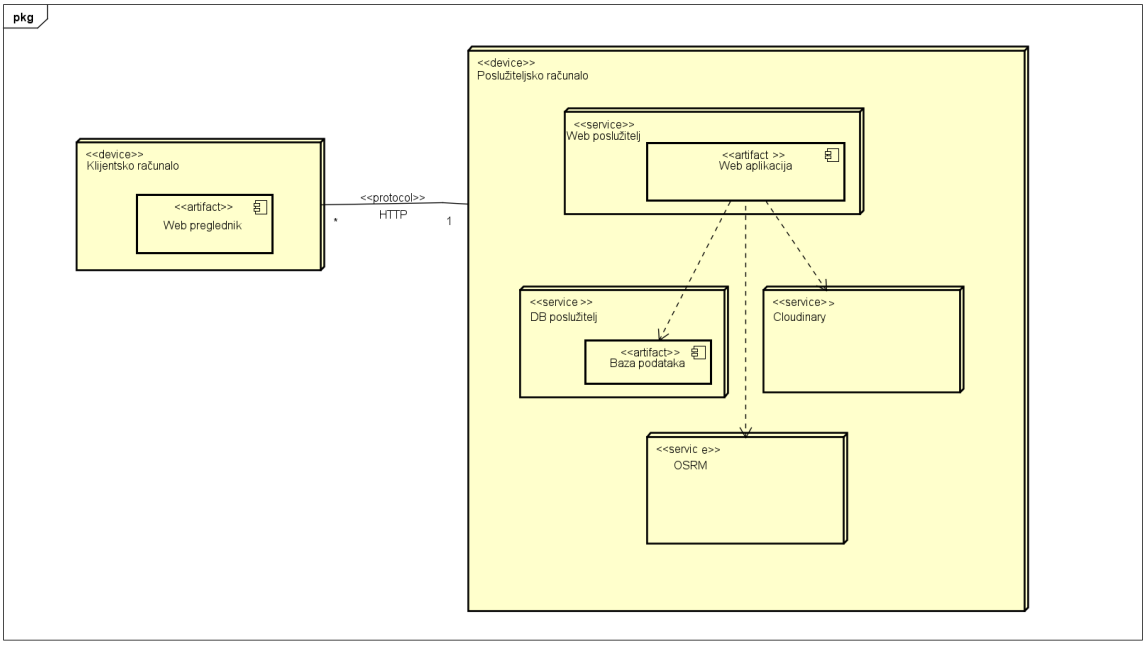
Ovo ispitivanje provodimo pomoću dodatka na Google Chrome *Selenium IDE*. Prijavimo se kao *igrac1*. Na ulaz testa unosimo podatak za kartografa (IBAN i slika osobne iskazice). Očekivani izlaz je prikaz zahtjeva kod *admin1*. Ove korake snimili smo pomoću Selenium IDE dodatka te smo dobili očekivani izlaz. Ukoliko u Selenium testu uklonimo jedan od podataka, željeni rezultat je da kod *admin1* nema zahtjeva za kartografa. Nakon pokretanja testa vidimo da test dobro radi, odnosno da kod *admin1* nema novih zahtjeva. Aplikacija je **prošla** test.



Slika 5.13: Selenium IDE, ispitivanje zahtjeva za kartografa

5.3 Dijagram razmještaja

Dijagrami razmještaja (engl. deployment diagrams) opisuju topologiju skopovlja i programsku potporu koja se koristi u implementaciji sustava u njegovom radnom i produkcijskom okruženju. Dijagram razmještaja prikazuje razvijene aplikacije. Na poslužiteljskom računalu nalaze se web poslužitelj, poslužitelj baze podataka te poslužitelji Cloudinary i OSRM. Kako bi pristupili web aplikaciji, klijenti koriste web preglednik. Razmjena podataka između korisnika (igrač, kartograf, admin) i poslužitelja odvija se korištenjem HTTP protokola.



Slika 5.14: Dijagram razmještaja

5.4 Upute za puštanje u pogon

Uvod

Aplikaciju je moguće lokalno pokrenuti tako da koristi H2 *in-memory* poslužitelj baze podatka ili PostgreSQL poslužitelj. Najprije je opisan postupak instalacije i konfiguracije PostgreSQL poslužitelja. Ti koraci mogu se preskočiti jer su nakon toga dane upute za pokretanje pomoću *docker-compose* skripte, čime je cijeli postupak značajno ubrzan i olakšan. Nakon toga opisan je način pokretanja Spring Boot aplikacije (*backend* dio) i React aplikacije (*frontend* dio.)

Instalacija PostgreSQL poslužitelja baze podataka

Potrebno je preuzeti PostgreSQL bazu podataka za odabrani operacijski sustav. U ovisnosti o operacijskom sustavu, potrebno je provesti standardnu instalaciju.

Konfiguracija PostgreSQL poslužitelja baze podataka

Poslužitelj baze podataka treba biti namješten da koristi port 5432. Potrebno je postaviti korisnika `postgres` s lozinkom `root`. Također treba pripremiti praznu bazu podataka naziva `geofighter_db`. Alternativni način postavljanja uključuje promjenu očekivanih parametara u datoteci *application-pg.properties* koja se nalazi u direktoriju `backend/src/main/resources`.

Pokretanje pomoću *docker-compose.yml* skripte

U slučaju korištenja usluge *Docker*, ovo je najjednostaviji način pokretanja PostgreSQL poslužitelja baze podataka. Iz komandne linije potrebno je pozicionirati se u direktorij *backend* i pokrenuti naredbu `docker-compose up`. Svi potrebni parametri PostgreSQL poslužitelja bit će postavljeni na vrijednosti koje aplikacija očekuje.

Pokretanje Spring Boot aplikacije

Za pokretanje Spring Boot aplikacije potrebno je, iz komandne linije, smjestiti se u direktorij *backend* i pokrenuti naredbu `mvnw spring-boot:run -Dspring-boot.run.profiles=<profil>` (Windows) ili `./mvnw spring-boot:run -Dspring-boot.run.profiles=<profil>` (Linux). Odabrani profil može biti `pg` (PostgreSQL) ili `h2` (H2), ovisno o preferiranom poslužitelju. Ako se koristi profil `pg`, potrebno je

najprije provesti ranije navedene korake za postavljanje PostgreSQL poslužitelja. Za profil `h2` nije ništa potrebno namještati.

Ako navedene naredbe za postavljanje profila ne rade, valja pokušati s `mvnw spring-boot:run -Dspring.profiles.active=<profil>` (Windows) ili `./mvnw spring-boot:run -Dspring.profiles.active=<profil>` (Linux). Profili pri tome poprimaju istu vrijednosti (`pg` ili `h2`).

Aplikacija će nakon pokretanja biti dostupna na portu 8080, a baza podataka će inicijalno biti popunjena podatcima zadanima u `data-<profil>.sql` skripti, pri čemu je profil `pg` ili `h2`.

Kako bi se aplikacija mogla ispravno koristiti, potrebno je postaviti očekivane varijable okruženja. To su: EMAIL, EMAIL_PASSWORD i CLOUDINARY_URL. Varijable EMAIL i EMAIL_PASSWORD predstavljaju podatke e-mail računa koji se koristi za slanje e-mail potvrde igračima koji su se tek registrirali. Varijabla CLOUDINARY_URL je personalizirana veza na vanjsku uslugu *Cloudinary* na koju se pohranjuju slike.

Pokretanje React aplikacije

Za pokretanje React aplikacije potrebno je instalirati okruženje *Node.js*, zajedno s upraviteljem paketa *npm*. Prije pokretanja React aplikacije potrebno je napraviti prethodni korak (pokretanje Spring Boot aplikacije). Zatim je potrebno pozicionirati se u direktorij *frontend* i najprije pokrenuti naredbu `npm install`, a zatim `npm start`. Time će se aplikacija postati dostupna na portu 3000, a automatski će se otvoriti `localhost:3000` u prozoru zadano postavljenog web preglednika.

6. Zaključak i budući rad

Zadatak naše grupe bio je razvoj igre *GeoFighter* u kojoj se igrači međusobno bore kartama. Igrači karte skupljaju na različitim stvarnim lokacijama, a cilj razvoja igre je popunjavanje manjkave baze podataka stvarnih lokacija. Nakon 15 tjedana timskog rada u razvijanju igre, uspješno smo ostvarili zadani cilj. Provedba projekta izvodila se kroz dvije faze.

Prva faza projekta uključivala je okupljanje tima za razvoj igrice, dodjele projektnih zadataka, formiranje podtimova za rad na *frontendu* i *backendu* te dokumentiranje zahtjeva igre. Kvalitetnim dokumentiranjem tijekom prve faze projekta uvelike smo olakšali daljnju implementaciju i rad na samoj igri te osmišljavanju sustava. Izrađeni obrasci i dijagrami (obrasci uporabe, sekvencijski dijagrami, model bazepodataka, dijagram razreda) pomogli su timovima za razvoj *frontenda* i *backenda* koje smo definirali u ranijem dijelu prve faze te su uštedili mnogo vremena tijekom druge faze projekta kada su članovi tima nailazili na nedoumice oko implementacije rješenja.

Druga faza projekta uglavnom se temeljila na samostalnom radu članova te na stjecanju novih znanja u slučaju manjka iskustva pri korištenju potrebnih odabranih alata i programskih jezika za izradu implementacijskih rješenja. Također, u drugoj fazi bilo je potrebno dovršiti i dokumentirati ostale UML dijagrame i napisati ostatak dokumentacije projekta. Temeljito dokumentirano željeno ponašanje sustava te dobro izrađeni obrasci i dijagrami tijekom prve faze projekta omogućili su nam da izbjegnemo potencijalne pogreške tijekom razvoja igre koje bi bile vremenski skupe u daljnjoj izradi projekta.

Dodatno smo proširili funkcionalnost igre dodavši mogućnost slanja poruka igračima. Još moguće proširenje postojeće inačice sustava je izrada mobilne aplikacije čime bi korisničko iskustvo bilo bogatije i bolje nego ono ostvareno web aplikacijom. Moguće su nadogradnje u obliku organiziranih turnira i dodavanju mogućnosti za proširenje udaljenosti za borbu kao i dodavanje različitih tipova borbi.

Sudjelovanje na ovom projektu naučilo nas je koliko znači dobra komunikacija među članovima tima, koliko je nužno međusobno poštivanje i razumijevanje

prema kolegama u timu te važnost dobre vremenske organiziranosti i koordiniranosti među članovima. Naposljetku, iako zadovoljni postignutim rezultatima i stečenim iskustvima tijekom rada na projektnom zadatku, svjesni smo potencijalnih nadogradnji kao i onih dijelova koje smo mogli bolje implementirati. Bez obzira na to, puno smo naučili.

Popis literature

1. Programsko inženjerstvo, FER ZEMRIS, http://www.fer.hr/predmet/proinz
2. Astah Community, http://astah.net/editions/uml-new
3. Spring Boot, https://spring.io/
4. React, https://reactjs.org
5. Java and Databases, https://www.marcobehler.com/guides/java-databases
6. EDRPlus, https://erdplus.com
7. DataGrip, https://www.jetbrains.com/datagrip/
8. Spring Data JPA, https://spring.io/projects/spring-data-jpa
9. Eclipse, https://www.eclipse.org
10. Visual Studio Code, https://code.visualstudio.com

Indeks slika i dijagrama

Dodatak: Prikaz aktivnosti grupe

Dnevnik sastajanja

1. sastanak
    - Datum: 6. listopada 2020.
    - Prisustvovali: M.Frandolić, L.Bašić, L.Brečić, P.Kopić, I.Krivačić, N.Petrović
    - Teme sastanka:
        - sastanak s asistentom i demonstratorom
        - analiza zadatka
        - raščišćavanje osnovnih dilema funkcionalnosti
        - okviran odabir alata i tehnologija
2. sastanak
    - Datum: 10. listopada 2020.
    - Prisustvovali: M.Frandolić, L.Bašić, L.Brečić, P.Kopić, I.Krivačić, N.Petrović
    - Teme sastanka:
        - konačan odabir alata i tehnologija
        - definiranje opisa projektnog zadatka
        - definiranje funkcionalnih zahtjeva
        - podjela opisa obrazaca uporabe
3. sastanak
    - Datum: 19. listopada 2020.
    - Prisustvovali: M.Frandolić, I.Krivačić
    - Teme sastanka:
        - kratko predavanje demosa na temu CI/CD

4. sastanak
    - Datum: 21. listopada 2020.
    - Prisustvovali: M.Frandolić, L.Bašić, L.Brečić, P.Kopić, I.Krivačić, N.Petrović
    - Teme sastanka:
        - sastanak s asistentom - dikusija o postojećim sličnim aplikacijama, potencijalnoj koristi projekta, moguće nadogradnje projektnog zadatka te rješavanje nejasnoća oko obrazaca uporabe
        - podjela zadatka(dijagrami obrazaca uporabe, sekvencijski dijagrami, baza podataka)
5. sastanak
    - Datum: 30. listopada 2020.
    - Prisustvovali: M.Frandolić, I.Krivačić
    - Teme sastanka:
        - sastanak s asistentom - diskusija o ispravnosti i nadopuni dijagarma obrazaca i sekvencijskih dijagrama
6. sastanak
    - Datum: 31. listopada 2020.
    - Prisustvovali: M.Frandolić, L.Bašić, L.Brečić, P.Kopić, I.Krivačić, N.Petrović
    - Teme sastanka:
        - podjela zadataka za frontend (početna stranica i mogućnost registracije), backend (funkcionalnost za registraciju, povezivanje s frontendom)
        - podjela posla za bazu podataka
7. sastanak
    - Datum: 6. studenoga 2020.
    - Prisustvovali: M.Frandolić, L.Bašić, L.Brečić, P.Kopić, N.Petrović
    - Teme sastanka:
        - sastanak s asistentom
        - rasprava o bazi podataka

8. sastanak
    - Datum: 7. studenoga 2020.
    - Prisustvovali: M.Frandolić, L.Bašić, L.Brečić, P.Kopić, I.Krivačić, N.Petrović
    - Teme sastanka:
        - rasprava o implementaciji baze podataka
        - raspravljanje o napravljenom frontendu i backendu
        - daljnja podjela posla za frontend(prikaz početnih stranica za igrača, admina i kartografa)
9. sastanak
    - Datum: 11. studenoga 2020.
    - Prisustvovali: M.Frandolić, L.Bašić, L.Brečić, P.Kopić, I.Krivačić, N.Petrović
    - Teme sastanka:
        - sastanak s asistentom i demonstratorom - demonstracija dosadašnjeg rada
        - podjela preostale dokumentacije
10. sastanak
    - Datum: 9. prosinca 2020.
    - Prisustvovali: M.Frandolić, L.Bašić, L.Brečić, P.Kopić, I.Krivačić, N.Petrović
    - Teme sastanka:
        - podjela u timove frontend i backend
        - podjela zadataka
11. sastanak
    - Datum: 11. prosinca 2020.
    - Prisustvovali: M.Frandolić, L.Brečić, P.Kopić, I.Krivačić
    - Teme sastanka:
        - dogovor oko daljnje podjele zadataka
12. sastanak
    - Datum: 13. prosinca 2020.
    - Prisustvovali: M.Frandolić, L.Brečić, N. Petrović
    - Teme sastanka:
        - dogovor oko dijelova zadataka backenda
13. sastanak
    - Datum: 15. prosinca 2020.
    - Prisustvovali: L. Bašič, P. Kopić, I. Krivačić
    - Teme sastanka:

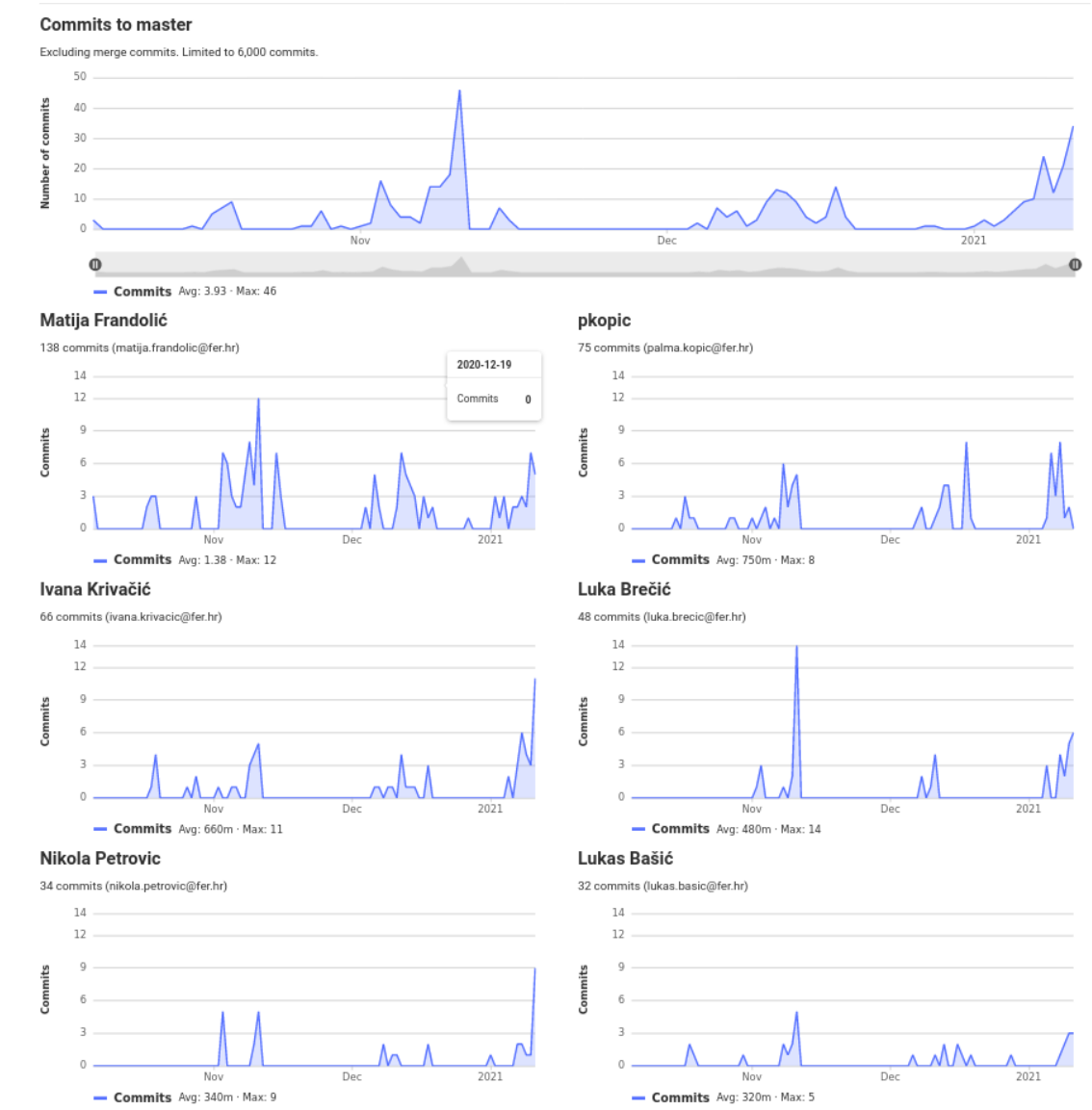
  - podjela poslova na frontendu

14. sastanak
    - Datum: 20. prosinca 2020.
    - Prisustvovali: M.Frandolić, L.Bašić, L.Brečić, P.Kopić, I.Krivačić, N.Petrović
    - Teme sastanka:
      - pregled napravljenog
      - raspravljanje oko nastalih nejasnoća
15. sastanak
    - Datum: 22. prosinca 2020.
    - Prisustvovali: M.Frandolić, L.Bašić, L.Brečić, P.Kopić, I.Krivačić, N.Petrović
    - Teme sastanka:
      - sastanak s asistentom - demonstracija alfa inačice
16. sastanak
    - Datum: 7. siječnja 2021.
    - Prisustvovali: M.Frandolić, L.Bašić, L.Brečić, P.Kopić, I.Krivačić, N.Petrović
    - Teme sastanka:
      - plan za daljnji period
      - podjela rada implementacije
17. sastanak
    - Datum: 10. siječnja 2021.
    - Prisustvovali: M.Frandolić, L.Bašić, L.Brečić, P.Kopić, I.Krivačić, N.Petrović
    - Teme sastanka:
      - podjela preostale implementacije
      - dovršavanje pisanje dokumenacije

Tablica aktivnosti

	Matija Frandolić	**Lukas Bašić**	**Luka Brečić**	**Palma Kopić**	**Ivana Krivačić**	**Nikola Petrović**	**Ozren Skerlev**
Upravljanje projektom	20						
Opis projektnog zadatka				9			
Funkcionalni zahtjevi		3					
Opis pojedinih obrazaca		2			6		
Dijagram obrazaca					4		
Sekvencijski dijagrami		2			4		
Opis ostalih zahtjeva					1		
Arhitektura i dizajn sustava			2	1	2		
Baza podataka			16			12	
Dijagram razreda			19			1	
Dijagram stanja					3		
Dijagram aktivnosti						5	
Dijagram komponenti				4			
Korištene tehnologije i alati						3	
Ispitivanje programskog rješenja			6		4		
Dijagram razmještaja					1		
Upute za puštanje u pogon	1						
Dnevnik sastajanja			2		2		
Zaključak i budući rad			2				
Popis literature					1		
izrada baze podataka			10			2	
spajanje s bazom podataka	2						
backend	50	5	35			38	
frontend	25	75		60	50		

Dijagrami pregleda promjena



Slika 6.1: Dijagrami pregleda promjena